Maximum Metric Spanning Tree made Byzantine Tolerant

Swan Dubois, Toshimitsu Masuzawa, Sébastien Tixeuil

Maximum Metric Spanning Tree made Byzantine Tolerant

Swan Dubois* Toshimitsu Masuzawa† Sébastien Tixeuil‡

Abstract

Self-stabilization is a versatile approach to fault-tolerance since it permits a distributed system to recover from any transient fault that arbitrarily corrupts the contents of all memories in the system. Byzantine tolerance is an attractive feature of distributed systems that permits to cope with arbitrary malicious behaviors. This paper focus on systems that are both self-stabilizing and Byzantine tolerant.

We consider the well known problem of constructing a maximum metric tree in this context. Combining these two properties is known to induce many impossibility results. In this paper, we provide first two impossibility results about the construction of maximum metric tree in presence of transients and (permanent) Byzantine faults. Then, we provide a new self-stabilizing protocol that provides optimal containment of an arbitrary number of Byzantine faults.

Keywords Byzantine fault, Distributed protocol, Fault tolerance, Stabilization, Spanning tree construction

1 Introduction

The advent of ubiquitous large-scale distributed systems advocates that tolerance to various kinds of faults and hazards must be included from the very early design of such systems. *Self-stabilization* [2, 3, 16] is a versatile technique that permits forward recovery from any kind of *transient* faults, while *Byzantine Fault-tolerance* [12] is traditionally used to mask the effect of a limited number of *malicious* faults. Making distributed systems tolerant to both transient and malicious faults is appealing yet proved difficult [4, 1, 15] as impossibility results are expected in many cases.

Related Works A promizing path towards multitolerance to both transient and Byzantine faults is *Byzantine containment*. For *local* tasks (*i.e.* tasks whose correctness can be checked locally, such as vertex coloring, link coloring, or dining philosophers), the notion of *strict stabilization* was proposed [15, 14]. Strict stabilization guarantees that there exists a *containment radius* outside which the effect of permanent faults is masked, provided that the problem specification makes it possible to break the causality chain that is caused by the faults. As many problems are not local, it turns out that it is impossible to provide strict stabilization for those. To circumvent impossibility results, the weaker notion of *strong stabilization* was proposed [13, 7]: here, correct nodes outside the containment radius may be perturbated by the actions of Byzantine node, but only a finite number of times.

*UPMC Sorbonne Universités & INRIA, France, swan.dubois@lip6.fr
†Osaka University, Japan, masuzawa@ist.osaka-u.ac.jp
‡UPMC Sorbonne Universités & Institut Universitaire de France, France, sebastien.tixeuil@lip6.fr

Recently, the idea of generalizing strict and strong stabilization to an area that depends on the graph topology and the problem to be solved rather than an arbitrary fixed containment radius was proposed [5, 6] and denoted by *topology aware* strict (and strong) stabilization. When maximizable metric trees are considered, [5] proposed an optimal (with respect to impossibility results) protocol for topology-aware strict stabilization, and for the simpler case of breath-first-search metric trees, [6] presented a protocol that is optimal both with respect to strict and strong variants of topology-aware stabilization. The case of optimality for topology-aware strong stabilization in the general maximal metric case remains open.

Our Contribution In this paper, we investigate the possibility of topology-aware strong stabilization for tasks that are global (*i.e.* for with there exists a causality chain of size r, where r depends on n the size of the network), and focus on the maximum metric tree problem. Our contribution in this paper is threefold. First, we provide two impossibility results for self-stabilizing maximum metric tree construction in presence of Byzantine faults. In more details, we characterize a specific class of maximizable metrics (which includes breath-first-search and shortest path metrics) that prevents the existence of strong stabilizing solutions and we generalize an impossibilty result of [6] that provides a lower bound on the containmemt area for topology-aware strong stabilization (Section 3). Second, we provide a topology-aware strongly stabilizing protocol that matches this lower bound on the containment area (Section 4). Finally, we provide a necessary and sufficient condition for the existence of a strongly stabilizing solution (Section 5).

2 Model, Definitions and Previous Results

2.1 State Model

A *distributed system* $S = (V, L)$ consists of a set $V = \{v_1, v_2, \ldots, v_n\}$ of processes and a set L of bidirectional communication links (simply called links). A link is an unordered pair of distinct processes. A distributed system S can be regarded as a graph whose vertex set is V and whose link set is L, so we use graph terminology to describe a distributed system S. We use the following notations: $n = |V|$, $m = |L|$ and $d(u, v)$ denotes the distance between two processes u and v (*i.e* the length of the shortest path between u and v).

Processes u and v are called *neighbors* if $(u, v) \in L$. The set of neighbors of a process v is denoted by N_v. We do not assume existence of a unique identifier for each process. Instead we assume each process can distinguish its neighbors from each other by locally labeling them.

In this paper, we consider distributed systems of arbitrary topology. We assume that a single process is distinguished as a *root*, and all the other processes are identical. We adopt the *shared state model* as a communication model in this paper, where each process can directly read the states of its neighbors.

The variables that are maintained by processes denote process states. A process may take actions during the execution of the system. An action is simply a function that is executed in an atomic manner by the process. The action executed by each process is described by a finite set of guarded actions of the form $\langle \text{guard} \rangle \longrightarrow \langle \text{statement} \rangle$. Each guard of process u is a boolean expression involving the variables of u and its neighbors.

A global state of a distributed system is called a *configuration* and is specified by a product of states of all processes. We define C to be the set of all possible configurations of a distributed system S. For a process set $R \subseteq V$ and two configurations ρ and ρ', we denote $\rho \overset{R}{\mapsto} \rho'$ when ρ changes to ρ' by executing an action of each process in R simultaneously. Notice that ρ and ρ'

can be different only in the states of processes in R. For completeness of execution semantics, we should clarify the configuration resulting from simultaneous actions of neighboring processes. The action of a process depends only on its state at ρ and the states of its neighbors at ρ, and the result of the action reflects on the state of the process at ρ'.

We say that a process is *enabled* in a configuration ρ if the guard of at least one of its actions is evaluated as true in ρ.

A *schedule* of a distributed system is an infinite sequence of process sets. Let $Q = R^1, R^2, \ldots$ be a schedule, where $R^i \subseteq V$ holds for each i ($i \geq 1$). An infinite sequence of configurations $e = \rho_0, \rho_1, \ldots$ is called an *execution* from an initial configuration ρ_0 by a schedule Q, if e satisfies $\rho_{i-1} \overset{R^i}{\mapsto} \rho_i$ for each i ($i \geq 1$). Process actions are executed atomically, and we distinguish some properties on the scheduler (or daemon). A *distributed daemon* schedules the actions of processes such that any subset of processes can simultaneously execute their actions. We say that the daemon is *central* if it schedules action of only one process at any step. The set of all possible executions from $\rho_0 \in C$ is denoted by E_{ρ_0}. The set of all possible executions is denoted by E, that is, $E = \bigcup_{\rho \in C} E_\rho$. We consider *asynchronous* distributed systems but we add the following assumption on schedules: any schedule is strongly fair (that is, it is impossible for any process to be infinitely often enabled without executing its action in an execution) and k-bounded (that is, it is impossible for any process to execute more than k actions between two consecutive action executions of any other process).

In this paper, we consider (permanent) *Byzantine faults*: a Byzantine process (*i.e.* a Byzantine-faulty process) can make arbitrary behavior independently from its actions. If v is a Byzantine process, v can repeatedly change its variables arbitrarily. For a given execution, the number of faulty processes is arbitrary but we assume that the root process is never faulty.

2.2 Self-Stabilizing Protocols Resilient to Byzantine Faults

Problems considered in this paper are so-called *static problems*, *i.e.* they require the system to find static solutions. For example, the spanning-tree construction problem is a static problem, while the mutual exclusion problem is not. Some static problems can be defined by a *specification predicate* (shortly, specification), $spec(v)$, for each process v: a configuration is a desired one (with a solution) if every process satisfies $spec(v)$. A specification $spec(v)$ is a boolean expression on variables of P_v ($\subseteq V$) where P_v is the set of processes whose variables appear in $spec(v)$. The variables appearing in the specification are called *output variables* (shortly, *O-variables*). In what follows, we consider a static problem defined by specification $spec(v)$.

A *self-stabilizing protocol* ([2]) is a protocol that eventually reaches a *legitimate configuration*, where $spec(v)$ holds at every process v, regardless of the initial configuration. Once it reaches a legitimate configuration, every process never changes its O-variables and always satisfies $spec(v)$. From this definition, a self-stabilizing protocol is expected to tolerate any number and any type of transient faults since it can eventually recover from any configuration affected by the transient faults. However, the recovery from any configuration is guaranteed only when every process correctly executes its action from the configuration, *i.e.*, we do not consider existence of permanently faulty processes.

When (permanent) Byzantine processes exist, Byzantine processes may not satisfy $spec(v)$. In addition, correct processes near the Byzantine processes can be influenced and may be unable to satisfy $spec(v)$. Nesterenko and Arora [15] define a *strictly stabilizing protocol* as a self-stabilizing protocol resilient to unbounded number of Byzantine processes.

Given an integer c, a *c-correct process* is a process defined as follows.

Definition 1 (c-correct process) *A process is c-correct if it is correct (*i.e. *not Byzantine)*

and located at distance more than c from any Byzantine process.

Definition 2 ((c, f)-containment) *A configuration ρ is* (c, f)-contained *for specification spec if, given at most f Byzantine processes, in any execution starting from ρ, every c-correct process v always satisfies spec(v) and never changes its O-variables.*

The parameter c of Definition 2 refers to the *containment radius* defined in [15]. The parameter f refers explicitly to the number of Byzantine processes, while [15] dealt with unbounded number of Byzantine faults (that is $f \in \{0 \dots n\}$).

Definition 3 ((c, f)-strict stabilization) *A protocol is* (c, f)-strictly stabilizing *for specification spec if, given at most f Byzantine processes, any execution $e = \rho_0, \rho_1, \dots$ contains a configuration ρ_i that is (c, f)-contained for spec.*

An important limitation of the model of [15] is the notion of *r-restrictive* specifications. Intuitively, a specification is r-restrictive if it prevents combinations of states that belong to two processes u and v that are at least r hops away. An important consequence related to Byzantine tolerance is that the containment radius of protocols solving those specifications is at least r. For some (global) problems r can not be bounded by a constant. In consequence, we can show that there exists no $(c, 1)$-strictly stabilizing protocol for such a problem for any (finite) integer c.

Strong stabilization To circumvent such impossibility results, [7] defines a weaker notion than the strict stabilization. Here, the requirement to the containment radius is relaxed, *i.e.* there may exist processes outside the containment radius that invalidate the specification predicate, due to Byzantine actions. However, the impact of Byzantine triggered action is limited in times: the set of Byzantine processes may only impact processes outside the containment radius a bounded number of times, even if Byzantine processes execute an infinite number of actions.

In the following of this section, we recall the formal definition of strong stabilization adopted in [7]. From the states of c-correct processes, *c-legitimate configurations* and *c-stable configurations* are defined as follows.

Definition 4 (c-legitimate configuration) *A configuration ρ is c-legitimate for* spec *if every c-correct process v satisfies spec(v).*

Definition 5 (c-stable configuration) *A configuration ρ is c-stable if every c-correct process never changes the values of its O-variables as long as Byzantine processes make no action.*

Roughly speaking, the aim of self-stabilization is to guarantee that a distributed system eventually reaches a c-legitimate and c-stable configuration. However, a self-stabilizing system can be disturbed by Byzantine processes after reaching a c-legitimate and c-stable configuration. The *c-disruption* represents the period where c-correct processes are disturbed by Byzantine processes and is defined as follows

Definition 6 (c-disruption) *A portion of execution $e = \rho_0, \rho_1, \dots, \rho_t$ $(t > 1)$ is a c-disruption if and only if the following holds:*

1. *e is finite,*
2. *e contains at least one action of a c-correct process for changing the value of an O-variable,*

3. ρ_0 *is c-legitimate for* spec *and c-stable, and*

4. ρ_t *is the first configuration after* ρ_0 *such that* ρ_t *is c-legitimate for* spec *and c-stable.*

Now we can define a self-stabilizing protocol such that Byzantine processes may only impact processes outside the containment radius a bounded number of times, even if Byzantine processes execute an infinite number of actions.

Definition 7 ((t, k, c, f)-time contained configuration) *A configuration* ρ_0 *is* (t, k, c, f)*-time contained for* spec *if given at most* f *Byzantine processes, the following properties are satisfied:*

1. ρ_0 *is c-legitimate for* spec *and c-stable,*

2. *every execution starting from* ρ_0 *contains a c-legitimate configuration for* spec *after which the values of all the O-variables of c-correct processes remain unchanged (even when Byzantine processes make actions repeatedly and forever),*

3. *every execution starting from* ρ_0 *contains at most* t *c-disruptions, and*

4. *every execution starting from* ρ_0 *contains at most* k *actions of changing the values of O-variables for each c-correct process.*

Definition 8 ((t, c, f)-strongly stabilizing protocol) *A protocol* A *is* (t, c, f)*-strongly stabilizing if and only if starting from any arbitrary configuration, every execution involving at most* f *Byzantine processes contains a* (t, k, c, f)*-time contained configuration that is reached after at most* l *rounds. Parameters* l *and* k *are respectively the* (t, c, f)*-stabilization time and the* (t, c, f)*-process-disruption times of* A*.*

Note that a (t, k, c, f)-time contained configuration is a (c, f)-contained configuration when $t = k = 0$, and thus, (t, k, c, f)-time contained configuration is a generalization (relaxation) of a (c, f)-contained configuration. Thus, a strongly stabilizing protocol is weaker than a strictly stabilizing one (as processes outside the containment radius may take incorrect actions due to Byzantine influence). However, a strongly stabilizing protocol is stronger than a classical self-stabilizing one (that may never meet their specification in the presence of Byzantine processes).

The parameters t, k and c are introduced to quantify the strength of fault containment, we do not require each process to know the values of the parameters.

Topology-aware Byzantine resilience We saw previously that there exist a number of impossibility results on strict stabilization due to the notion of r-restrictive specifications. To circumvent this impossibility result, we describe here another weaker notion than the strict stabilization: the *topology-aware strict stabilization* (denoted by TA strict stabilization for short) introduced by [5]. Here, the requirement to the containment radius is relaxed, *i.e.* the set of processes which may be disturbed by Byzantine ones is not reduced to the union of c-neighborhood of Byzantine processes (*i.e.* the set of processes at distance at most c from a Byzantine process) but can be defined depending on the graph topology and Byzantine processes location.

In the following, we give formal definition of this new kind of Byzantine containment. From now, B denotes the set of Byzantine processes and S_B (which is function of B) denotes a subset of V (intuitively, this set gathers all processes which may be disturbed by Byzantine processes).

Definition 9 (S_B-correct node) *A node is* S_B-correct *if it is a correct node (*i.e. *not Byzantine) which not belongs to* S_B*.*

Definition 10 (S_B-legitimate configuration) *A configuration ρ is* S_B-legitimate *for spec if every S_B-correct node v is legitimate for spec (*i.e. *if spec(v) holds).*

Definition 11 ((S_B, f)-topology-aware containment) *A configuration ρ_0 is* (S_B, f)-topology-aware contained *for specification spec if, given at most f Byzantine processes, in any execution $e = \rho_0, \rho_1, \ldots$, every configuration is S_B-legitimate and every S_B-correct process never changes its O-variables.*

The parameter S_B of Definition 11 refers to the *containment area*. Any process which belongs to this set may be infinitely disturbed by Byzantine processes. The parameter f refers explicitly to the number of Byzantine processes.

Definition 12 ((S_B, f)-topology-aware strict stabilization) *A protocol is* (S_B, f)-topology-aware strictly stabilizing *for specification spec if, given at most f Byzantine processes, any execution $e = \rho_0, \rho_1, \ldots$ contains a configuration ρ_i that is (S_B, f)-topology-aware contained for spec.*

Note that, if B denotes the set of Byzantine processes and $S_B = \left\{ v \in V | \min_{b \in B} (d(v, b)) \leq c \right\}$, then a (S_B, f)-topology-aware strictly stabilizing protocol is a (c, f)-strictly stabilizing protocol. Then, the concept of topology-aware strict stabilization is a generalization of the strict stabilization. However, note that a TA strictly stabilizing protocol is stronger than a classical self-stabilizing protocol (that may never meet their specification in the presence of Byzantine processes). The parameter S_B is introduced to quantify the strength of fault containment, we do not require each process to know the actual definition of the set.

Similarly to topology-aware strict stabilization, we can weaken the notion of strong stabilization using the notion of containment area. This idea was introduced by [6]. We recall in the following the formal definition of this concept.

Definition 13 (S_B-stable configuration) *A configuration ρ is S_B-stable if every S_B-correct process never changes the values of its O-variables as long as Byzantine processes make no action.*

Definition 14 (S_B-TA-disruption) *A portion of execution $e = \rho_0, \rho_1, \ldots, \rho_t$ $(t > 1)$ is a S_B-TA-disruption if and only if the followings hold:*

1. *e is finite,*
2. *e contains at least one action of a S_B-correct process for changing the value of an O-variable,*
3. *ρ_0 is S_B-legitimate for spec and S_B-stable, and*
4. *ρ_t is the first configuration after ρ_0 such that ρ_t is S_B-legitimate for spec and S_B-stable.*

Definition 15 ((t, k, S_B, f)-TA time contained configuration) *A configuration ρ_0 is (t, k, S_B, f)-TA time contained for* spec *if given at most f Byzantine processes, the following properties are satisfied:*

1. *ρ_0 is S_B-legitimate for* spec *and S_B-stable,*

2. *every execution starting from ρ_0 contains a S_B-legitimate configuration for* spec *after which the values of all the O-variables of S_B-correct processes remain unchanged (even when Byzantine processes make actions repeatedly and forever),*

3. *every execution starting from ρ_0 contains at most t S_B-TA-disruptions, and*

4. *every execution starting from ρ_0 contains at most k actions of changing the values of O-variables for each S_B-correct process.*

Definition 16 ((t, S_B, f)-TA strongly stabilizing protocol) *A protocol A is (t, S_B, f)-TA strongly stabilizing if and only if starting from any arbitrary configuration, every execution involving at most f Byzantine processes contains a (t, k, S_B, f)-TA-time contained configuration that is reached after at most l rounds of each S_B-correct node. Parameters l and k are respectively the (t, S_B, f)-stabilization time and the (t, S_B, f)-process-disruption time of A.*

2.3 Maximum Metric Tree Construction

In this work, we deal with maximum (routing) metric trees as defined in [10]. Informally, the goal of a routing protocol is to construct a tree that simultaneously maximizes the metric values of all of the nodes with respect to some total ordering $\prec$. In the following, we recall all definitions and notations introduced in [10].

Definition 17 (Routing metric) *A* routing metric *(or just* metric*) is a five-tuple $(M, W, met, mr, \prec)$ where:*

1. *M is a set of metric values,*

2. *W is a set of edge weights,*

3. *met is a metric function whose domain is $M \times W$ and whose range is M,*

4. *mr is the maximum metric value in M with respect to $\prec$ and is assigned to the root of the system,*

5. *$\prec$ is a less-than total order relation over M that satisfies the following three conditions for arbitrary metric values m, m', and m'' in M:*

   (a) *irreflexivity: $m \not\prec m$,*

   (b) *transitivity : if $m \prec m'$ and $m' \prec m''$ then $m \prec m''$,*

   (c) *totality: $m \prec m'$ or $m' \prec m$ or $m = m'$.*

Any metric value $m \in M \backslash \{mr\}$ satisfies the utility condition *(that is, there exist $w_0, \ldots, w_{k-1}$ in W and $m_0 = mr, m_1, \ldots, m_{k-1}, m_k = m$ in M such that $\forall i \in \{1, \ldots, k\}, m_i = met(m_{i-1}, w_{i-1})$).*

For instance, we provide the definition of four classical metrics with this model: the shortest path metric ($\mathcal{SP}$), the flow metric ($\mathcal{F}$), and the reliability metric ($\mathcal{R}$). Note also that we can modelise the construction of a spanning tree with no particular constraints in this model using the metric $\mathcal{NC}$ described below and the construction of a BFS spanning tree using the shortest path metric ($\mathcal{SP}$) with $W_1 = \{1\}$ (we denoted this metric by $\mathcal{BFS}$ in the following).

$$
\begin{array}{rcl}
\mathcal{SP} & = & (M_1, W_1, met_1, mr_1, \prec_1) \\
\text{where} & & M_1 = \mathbb{N} \\
& & W_1 = \mathbb{N} \\
& & met_1(m, w) = m + w \\
& & mr_1 = 0 \\
& & \prec_1 \text{ is the classical } > \text{ relation}
\end{array}
\qquad
\begin{array}{rcl}
\mathcal{F} & = & (M_2, W_2, met_2, mr_2, \prec_2) \\
\text{where} & & mr_2 \in \mathbb{N} \\
& & M_2 = \{0, \ldots, mr_2\} \\
& & W_2 = \{0, \ldots, mr_2\} \\
& & met_2(m, w) = min\{m, w\} \\
& & \prec_2 \text{ is the classical } < \text{ relation}
\end{array}
$$

$$
\begin{array}{rcl}
\mathcal{R} & = & (M_3, W_3, met_3, mr_3, \prec_3) \\
\text{where} & & M_3 = [0, 1] \\
& & W_3 = [0, 1] \\
& & met_3(m, w) = m * w \\
& & mr_3 = 1 \\
& & \prec_3 \text{ is the classical } < \text{ relation}
\end{array}
\qquad
\begin{array}{rcl}
\mathcal{NC} & = & (M_4, W_4, met_4, mr_4, \prec_4) \\
\text{where} & & M_4 = \{0\} \\
& & W_4 = \{0\} \\
& & met_4(m, w) = 0 \\
& & mr_4 = 0 \\
& & \prec_4 \text{ is the classical } < \text{ relation}
\end{array}
$$

Definition 18 (Assigned metric) *An* assigned metric *over a system S is a six-tuple $(M, W, met, mr, \prec, wf)$ where $(M, W, met, mr, \prec)$ is a metric and wf is a function that assigns to each edge of S a weight in W.*

Let a rooted path (from v) be a simple path from a process v to the root r. The next set of definitions are with respect to an assigned metric $(M, W, met, mr, \prec, wf)$ over a given system S.

Definition 19 (Metric of a rooted path) *The* metric of a rooted path *in S is the prefix sum of met over the edge weights in the path and mr.*

For example, if a rooted path p in S is $v_k, \ldots, v_0$ with $v_0 = r$, then the metric of p is $m_k = met(m_{k-1}, wf(\{v_k, v_{k-1}\}))$ with $\forall i \in \{1, \ldots, k-1\}, m_i = met(m_{i-1}, wf(\{v_i, v_{i-1}\})$ and $m_0 = mr$.

Definition 20 (Maximum metric path) *A rooted path p from v in S is called a* maximum metric path *with respect to an assigned metric if and only if for every other rooted path q from v in S, the metric of p is greater than or equal to the metric of q with respect to the total order $\prec$.*

Definition 21 (Maximum metric of a node) *The* maximum metric of a node $v \neq r$ *(or simply* metric value *of v) in S is defined by the metric of a maximum metric path from v. The maximum metric of r is mr.*

Definition 22 (Maximum metric tree) *A spanning tree T of S is a* maximum metric tree *with respect to an assigned metric over S if and only if every rooted path in T is a maximum metric path in S with respect to the assigned metric.*

The goal of the work of [10] is the study of metrics that always allow the construction of a maximum metric tree. More formally, the definition follows.

Definition 23 (Maximizable metric) *A metric is* maximizable *if and only if for any assignment of this metric over any system S, there is a maximum metric tree for S with respect to the assigned metric.*

Given a maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$, the aim of this work is to study the construction of a maximum metric tree with respect to $\mathcal{M}$ which spans the system in a self-stabilizing way in a system subject to permanent Byzantine faults (but we must assume that the root process is never a Byzantine one). It is obvious that these Byzantine processes may disturb some correct processes. It is why we relax the problem in the following way: we want to construct a maximum metric forest with respect to $\mathcal{M}$. The root of any tree of this forest must be either the real root or a Byzantine process.

Each process v has three O-variables: a pointer to its parent in its tree ($prnt_v \in N_v \cup \{\perp\}$), a level which stores its current metric value ($level_v \in M$) and an integer which stores a distance ($dist_v \in \mathbb{N}$). Obviously, Byzantine process may disturb (at least) their neighbors. We use the following specification of the problem.

We introduce new notations as follows. Given an assigned metric $(M, W, met, mr, \prec, wf)$ over the system S and two processes u and v, we denote by $\mu(u, v)$ the maximum metric of node u when v plays the role of the root of the system. If u and v are neighbors, we denote by $w_{u,v}$ the weight of the edge $\{u, v\}$ (that is, the value of $wf(\{u, v\})$).

Definition 24 ($\mathcal{M}$-path) *Given an assigned metric $\mathcal{M} = (M, W, mr, met, \prec, wf)$ over a system S, a path $(v_0, \ldots, v_k)$ $(k \geq 1)$ of S is a* $\mathcal{M}$-path *if and only if:*

1. $prnt_{v_0} = \perp$, $level_{v_0} = mr$, $dist_{v_0} = 0$, *and* $v_0 \in B \cup \{r\}$,
2. $\forall i \in \{1, \ldots, k\}, prnt_{v_i} = v_{i-1}$ *and* $level_{v_i} = met(level_{v_{i-1}}, w_{v_i, v_{i-1}})$,
3. $\forall i \in \{1, \ldots, k\}, met(level_{v_{i-1}}, w_{v_i, v_{i-1}}) = \max\limits_{u \in N_v}{}_{\prec}\{met(level_u, w_{v_i, u})\}$,
4. $\forall i \in \{1, \ldots, k\}, dist_{v_i} = legal\_dist_{v_{i-1}}$ *with* $\forall u \in N_v, legal\_dist_u = \begin{cases} dist_u + 1 \text{ if } level_v = level_u \\ 0 \text{ otherwise} \end{cases}$, *and*
5. $level_{v_k} = \mu(v_k, v_0)$.

We define the specification predicate $spec(v)$ of the maximum metric tree construction with respect to a maximizable metric $\mathcal{M}$ as follows.

$$spec(v) : \begin{cases} prnt_v = \perp \text{ and } level_v = mr, \text{ and } dist_v = 0 \text{ if } v \text{ is the root } r \\ \text{there exists a } \mathcal{M}\text{-path } (v_0, \ldots, v_k) \text{ such that } v_k = v \text{ otherwise} \end{cases}$$

2.4 Previous results

In this section, we summarize known results about maximum metric tree construction. The first interesting result about maximizable metrics is due to [10] that provides a fully characterization of maximizable metrics as follow.

Definition 25 (Boundedness) *A metric $(M, W, met, mr, \prec)$ is* bounded *if and only if: $\forall m \in M, \forall w \in W, met(m, w) \prec m$ or $met(m, w) = m$*

Definition 26 (Monotonicity) *A metric $(M, W, met, mr, \prec)$ is* monotonic *if and only if: $\forall (m,\ m') \in M^2, \forall w \in W, m \prec m' \Rightarrow (met(m, w) \prec met(m', w)$ or $met(m, w) = met(m', w))$*

Theorem 1 (Characterization of maximizable metrics [10]) *A metric is maximizable if and only if this metric is bounded and monotonic.*

Secondly, [9] provides a self-stabilizing protocol to construct a maximum metric tree with respect to any maximizable metric. Now, we focus on self-stabilizating solutions resilient to Byzantine faults. Following discussion of Section 2, it is obvious that there exists no strictly stabilizing protocol for this problem. If we consider the weaker notion of topology-aware strict stabilization, [5] defines the best containment area as:

$$S_B = \{v \in V \setminus B \,|\mu(v,r) \preceq max_{\prec}\{\mu(v,b), b \in B\}\} \setminus \{r\}$$

Intuitively, S_B gathers correct processes that are closer (or at equal distance) from a Byzantine process than the root according to the metric. Moreover, [5] proves that the algorithm introduced for the maximum metric spanning tree construction in [9] performed this optimal containment area. More formally, [5] proves the following results.

Theorem 2 ([5]) *Given a maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$, even under the central daemon, there exists no $(A_B, 1)$-TA-strictly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{M}$ where $A_B \subsetneq S_B$.*

Theorem 3 ([5]) *Given a maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$, the protocol of [9] is a $(S_B, n-1)$-TA strictly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{M}$.*

Some other works try to circumvent the impossibility result of strict stabilization using the concept ot strong stabilization but do not provide results for any maximizable metric. Indeed, [7] proves the following result about spanning tree.

Theorem 4 ([7]) *There exists a $(t, 0, n-1)$-strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{NC}$ (that is, for a spanning tree with no particular constraints) with a finite t.*

On the other hand, regarding BFS spanning tree construction, [6] proved the following impossibility result.

Theorem 5 ([6]) *Even under the central daemon, there exists no $(t, c, 1)$-strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{BFS}$ where t and c are two finite integers.*

Now, if we focus on topology-aware strong stabilization, [6] introduced the following containment area: $S^*_B = \{v \in V | \min_{b \in B}(d(v,b)) < d(r,v)\}$, and proved the following results.

Theorem 6 ([6]) *Even under the central daemon, there exists no $(t, A^*_B, 1)$-TA strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{BFS}$ where $A^*_B \subsetneq S^*_B$ and t is a finite integer.*

Theorem 7 ([6]) *The protocol of [11] is a $(t, S^*_B, n-1)$-TA strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{BFS}$ where t is a finite integer.*

The main motivation of this work is to fill the gap between results about TA strong and strong stabilization in the general case (that is, for any maximizable metric). Mainly, we define the best possible containment area for TA strong stabilization, we propose a protocol that provides this containment area and we characterize the set of metrics that allow strong stabilization.

3 Impossibility Results

In this section, we provide our impossibility results about containment radius (respectively area) of any strongly stabilizing (respectively TA strongly stabilizing) protocol for the maximum metric tree construction.

3.1 Strong Stabilization

We introduce here some new definitions to characterize some important properties of maximizable metrics that are used in the following.

Definition 27 (Strictly decreasing metric) *A metric* $\mathcal{M} = (M, W, mr, met, \prec)$ *is* strictly decreasing *if, for any metric value* $m \in M$*, the following property holds: either* $\forall w \in W, met(m, w) \prec m$ *or* $\forall w \in W, met(m, w) = m$*.*

Definition 28 (Fixed point) *A metric value* m *is a* fixed point *of a metric* $\mathcal{M} = (M, W, mr, met, \prec)$ *if* $m \in M$ *and if for any value* $w \in W$*, we have:* $met(m, w) = m$*.*

Then, we define a specific class of maximizable metrics and we prove that it is impossible to construct a maximum metric tree in a strongly-stabilizing way if we do not consider such a metric.

Definition 29 (Strongly maximizable metric) *A maximizable metric* $\mathcal{M} = (M, W, mr, met, \prec)$ *is strongly maximizable if and only if* $|M| = 1$ *or if the following properties holds:*

- $|M| \geq 2$*,*
- $\mathcal{M}$ *is strictly decreasing, and*
- $\mathcal{M}$ *has one and only one fixed point.*

Note that $\mathcal{NC}$ is a strongly maximizable metric (since $|M_4| = 1$) whereas $\mathcal{BFS}$ or $\mathcal{SP}$ are not (since the first one has no fixed point, the second is not strictly decreasing). If we consider the metric $\mathcal{MET}$ defined below, we can show that $\mathcal{MET}$ is a strongly maximizable metric such that $|M| \geq 2$.

$$\begin{array}{lll} \mathcal{MET} & = & (M_5, W_5, met_5, mr_5, \prec_5) \\ \text{where} & & M_5 = \{0, 1, 2, 3\} \\ & & W_5 = \{1\} \\ & & met_5(m, w) = max\{0, m - w\} \\ & & mr_5 = 3 \\ & & \prec_5 \text{ is the classical } < \text{ relation} \end{array}$$

Now, we can state our first impossibility result.

Theorem 8 *Given a maximizable metric* $\mathcal{M} = (M, W, mr, met, \prec)$*, even under the central daemon, there exists no* $(t, c, 1)$*-strongly stabilizing protocol for maximum metric spanning tree construction with respect to* $\mathcal{M}$ *for any finite integer* t *if:*

$$\begin{cases} \mathcal{M} \text{ is not a strongly maximizable metric, or} \\ c < |M| - 2 \end{cases}$$

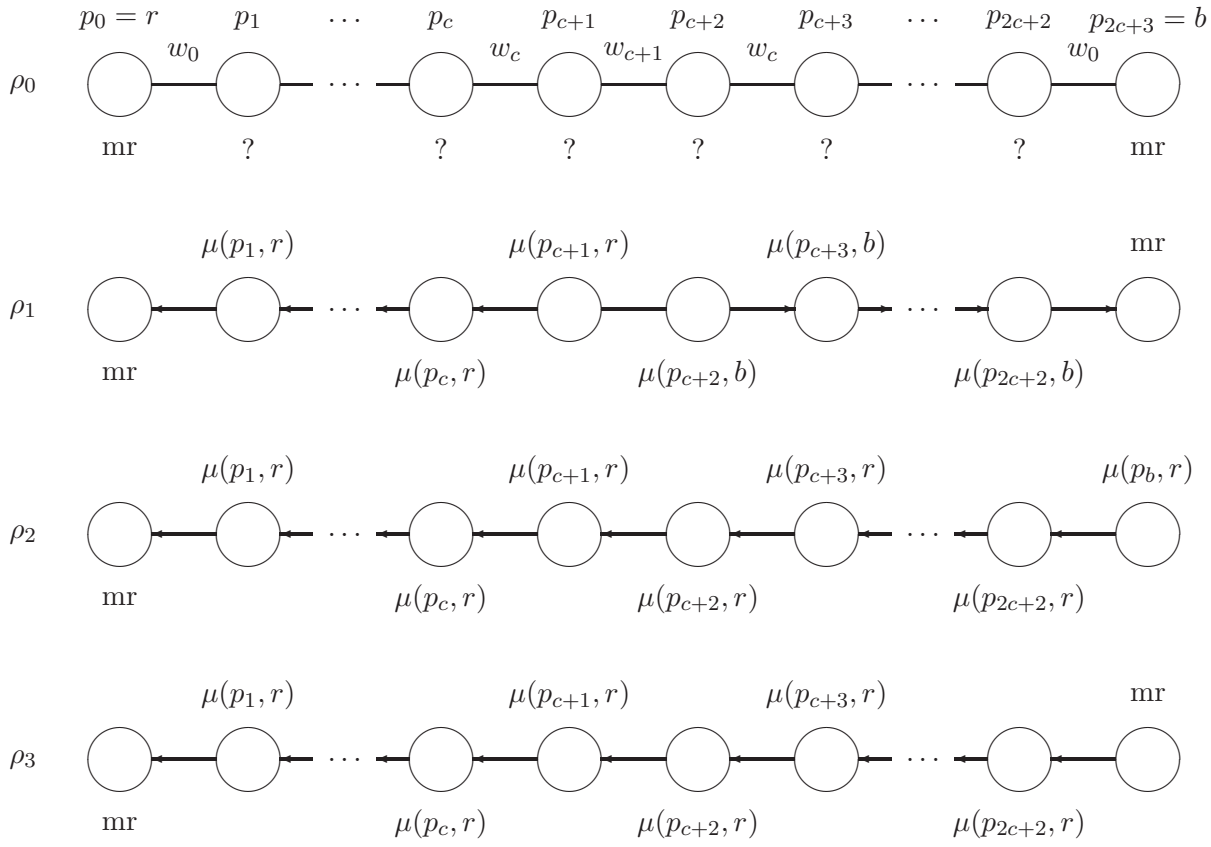

Figure 1: Configurations used in proof of Theorem 8, case 1.

Proof We prove this result by contradiction. We assume that $\mathcal{M} = (M, W, mr, met, \prec)$ is a maximizable metric such that there exist a finite integer t and a protocol $\mathcal{P}$ that is a $(t, c, 1)$-strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{M}$. We distinguish the following cases (note that they are exhaustive):

Case 1: $\mathcal{M}$ is a strongly maximizing metric and $c < |M| - 2$.

As $c \geq 0$, we know that $|M| \geq 2$ and by definition of a strongly stabilizing metric, $\mathcal{M}$ is strictly decreasing and has one and only one fixed point.

By assumption on $\mathcal{M}$, we know that there exist $c + 3$ distinct metric values $m_0 = mr, m_1, \ldots, m_{c+2}$ in M and $w_0, w_1, \ldots, w_{c+1}$ in W such that: $\forall i \in \{1, \ldots, c+2\}, m_i = met(m_{i-1}, w_{i-1}) \prec m_{i-1}$.

Let $S = (V, E, \mathcal{W})$ be the following weighted system $V = \{p_0 = r, p_1, \ldots, p_{2c+2}, p_{2c+3} = b\}$, $E = \{\{p_i, p_{i+1}\}, i \in \{0, \ldots, 2c+2\}\}$ and $\forall i \in \{0, c+1\}, w_{p_i,p_{i+1}} = w_{p_{2c+3-i},p_{2c+2-i}} = w_i$. Note that the choice $w_{p_{c+1},p_{c+2}} = w_{c+1}$ ensures us the following property when $level_r = level_b = mr$: $\mu(p_{c+1}, b) \prec \mu(p_{c+1}, r)$ (and by symmetry, $\mu(p_{c+2}, r) \prec \mu(p_{c+2}, b)$). Process p_0 is the real root and process b is a Byzantine one. Note that the construction of $\mathcal{W}$ ensures the following properties when $level_r = level_b = mr$: $\forall i \in \{1, \ldots, c+1\}, \mu(p_i, r) = \mu(p_{2c+3-i}, b)$, $\mu(p_i, b) \prec \mu(p_i, r)$ and $\mu(p_{2c+3-i}, r) \prec \mu(p_{2c+3-i}, b)$.

Assume that the initial configuration ρ_0 of S satisfies: $prnt_r = prnt_b = \perp$, $level_r =$

$level_b = mr$, and other variables of b (in particular $dist$) are identical to those of r (see Figure 1, variables of other processes may be arbitrary). Assume now that b takes exactly the same actions as r (if any) immediately after r. Then, by symmetry of the execution and by convergence of $\mathcal{P}$ to $spec$, we can deduce that the system reaches in a finite time a configuration ρ_1 (see Figure 1) in which: $\forall i \in \{1, \ldots, c+1\}, prnt_{p_i} = p_{i-1}$, $level_{p_i} = \mu(p_i, r) = m_i$, $dist_{p_i} = legal\_dist_{prnt_{p_i}}$ and $\forall i \in \{c+2, \ldots, 2c+2\}, prnt_{p_i} = p_{i+1}$, $level_{p_i} = \mu(p_i, b) = m_{2c+3-i}$, and $dist_{p_i} = legal\_dist_{prnt_{p_i}}$ (because this configuration is the only one in which all correct process v satisfies $spec(v)$ when $prnt_r = prnt_b = \perp$ and $level_r = level_b = mr$ by construction of $\mathcal{W}$). Note that ρ_1 is c-legitimate and c-stable.

Assume now that the Byzantine process acts as a correct process and executes correctly its algorithm. Then, by convergence of $\mathcal{P}$ in fault-free systems (remember that a strongly-stabilizing algorithm is a special case of self-stabilizing algorithm), we can deduce that the system reach in a finite time a configuration ρ_2 (see Figure 1) in which: $\forall i \in \{1, \ldots, 2c+3\}, prnt_{p_i} = p_{i-1}$, $level_{p_i} = \mu(p_i, r)$, and $dist_{p_i} = legal\_dist_{prnt_{p_i}}$ (because this configuration is the only one in which all process v satisfies $spec(v)$). Note that the portion of execution between ρ_1 and ρ_2 contains at least one c-perturbation (p_{c+2} is a c-correct process and modifies at least once its O-variables) and that ρ_2 is c-legitimate and c-stable.

Assume now that the Byzantine process b takes the following state: $prnt_b = \perp$ and $level_b = mr$. This step brings the system into configuration ρ_3 (see Figure 1). From this configuration, we can repeat the execution we constructed from ρ_0. By the same token, we obtain an execution of $\mathcal{P}$ which contains c-legitimate and c-stable configurations (see ρ_1) and an infinite number of c-perturbation which contradicts the $(t, c, 1)$-strong stabilization of $\mathcal{P}$.

Case 2: $\mathcal{M}$ is not strictly decreasing.

By definition, we know that $\mathcal{M}$ is not a strongly maximizable metric. Hence, we have $|M| \geq 2$. Then, the definition of a strictly decreasing metric implies that there exists a metric value $m \in M$ such that: $\exists w \in W$, $met(m, w) = m$ and $\exists w' \in W, m' = met(m, w') \prec m$ (and thus m is not a fixed point of $\mathcal{M}$). By the utility condition on M, we know that there exists a sequence of metric values $m_0 = mr, m_1, \ldots, m_l = m$ in M and $w_0, w_1, \ldots, w_{l-1}$ in W such that $\forall i \in \{1, \ldots, l\}, m_i = met(m_{i-1}, w_{i-1})$. Denote by k the length of the shortest such sequence. Note that this implies that $\forall i \in \{1, \ldots, k\}, m_i \prec m_{i-1}$ (otherwise we can remove m_i from the sequence and this is contradictory with the construction of k). We distinguish the following cases:

Case 2.1: $k \geq c + 2$.

We can use the same token as case 1 above by using w' instead of w_{c+1} in the case where $k = c + 2$ (since we know that $met(m, w') \prec m$).

Case 2.2: $k < c + 2$.

Let $S_1 = (V, E, \mathcal{W})$ be the following weighted system $V = \{p_0 = r, p_1, \ldots, p_{2c+2}, p_{2c+3} = b\}$, $E = \{\{p_i, p_{i+1}\}, i \in \{0, \ldots, 2c+2\}\}$, $\forall i \in \{0, \ldots, k-1\}, w_{p_i,p_{i+1}} = w_{p_{2c+3-i},p_{2c+2-i}} = w_i$, $\forall i \in \{k, \ldots, c\}, w_{p_i,p_{i+1}} = w_{p_{2c+3-i},p_{2c+2-i}} = w$ and $w_{p_{c+1},p_{c+2}} = w'$ (see Figure 2). Note that this choice ensures us the following property when $level_r = level_b = mr$: $\mu(p_{c+1}, b) \prec \mu(p_{c+1}, r)$ (and by symmetry, $\mu(p_{c+2}, r) \prec \mu(p_{c+2}, b)$). Process p_0 is the real root and process b is a Byzantine one. Note that the construction of $\mathcal{W}$ ensures the following properties when $level_r = level_b = mr$: $\forall i \in \{1, \ldots, c+1\}, \mu(p_i, r) = \mu(p_{2c+3-i}, b)$, $\mu(p_i, b) \prec \mu(p_i, r)$ and $\mu(p_{2c+3-i}, r) \prec \mu(p_{2c+3-i}, b)$.

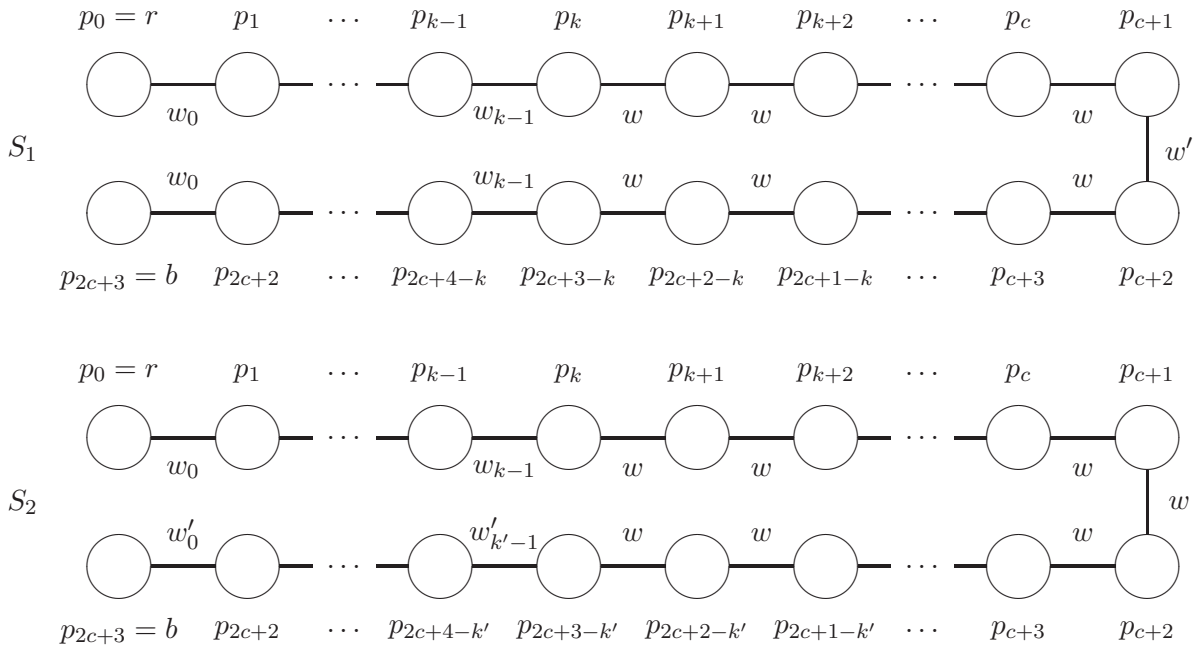

Figure 2: Configurations used in proof of Theorem 8, cases 2 and 3.

This construction allows us to follow the same proof as in case 1 above.

Case 3: $\mathcal{M}$ has no or more than two fixed point, and is strictly decreasing.

If $\mathcal{M}$ has no fixed point and is strictly decreasing, then $|M|$ is not finite and then, we can apply the result of case 1 above since c is a finite integer.

If $\mathcal{M}$ has two or more fixed points and is strictly decreasing, denote by Υ and Υ' two fixed points of $\mathcal{M}$. Without loss of generality, assume that $\Upsilon \prec \Upsilon'$. By the utility condition on M, we know that there exists sequences of metric values $m_0 = mr, m_1, \ldots, m_l = \Upsilon$ and $m'_0 = mr, m'_1, \ldots, m'_{l'} = \Upsilon'$ in M and $w_0, w_1, \ldots, w_{l-1}$ and $w'_0, w'_1, \ldots, w'_{l'-1}$ in W such that $\forall i \in \{1, \ldots, l\}, m_i = met(m_{i-1}, w_{i-1})$ and $\forall i \in \{1, \ldots, l'\}, m'_i = met(m'_{i-1}, w'_{i-1})$. Denote by k and k' the length of shortest such sequences. Note that this implies that $\forall i \in \{1, \ldots, k\}, m_i \prec m_{i-1}$ and $\forall i \in \{1, \ldots, k'\}, m'_i \prec m'_{i-1}$ (otherwise we can remove m_i or m'_i from the corresponding sequence). We distinguish the following cases:

Case 3.1: $k > c + 2$ or $k' > c + 2$.

Without loss of generality, assume that $k > c + 2$ (the second case is similar). We can use the same token as case 1 above.

Case 3.2: $k \leq c + 2$ and $k' \leq c + 2$.

Let w be an arbitrary value of W. Let $S_2 = (V, E, \mathcal{W})$ be the following weighted system $V = \{p_0 = r, p_1, \ldots, p_{2c+2}, p_{2c+3} = b\}$, $E = \{\{p_i, p_{i+1}\}, i \in \{0, \ldots, 2c + 2\}\}$, $\forall i \in \{0, k - 1\}, w_{p_i,p_{i+1}} = w_i$, $\forall i \in \{0, k' - 1\}, w_{p_{2c+3-i},p_{2c+2-i}} = w'_i$ and $\forall i \in \{k, 2c + 2 - k'\}, w_{p_i,p_{i+1}} = w$ (see Figure 2). Note that this choice ensures us the following property when $level_r = level_b = mr$: $\mu(p_{c+1}, r) = \Upsilon \prec \Upsilon' = \mu(p_{c+1}, b)$ and $\mu(p_{c+2}, r) = \Upsilon \prec \Upsilon' = \mu(p_{c+2}, b)$. Process p_0 is the real root and process b is a Byzantine one.

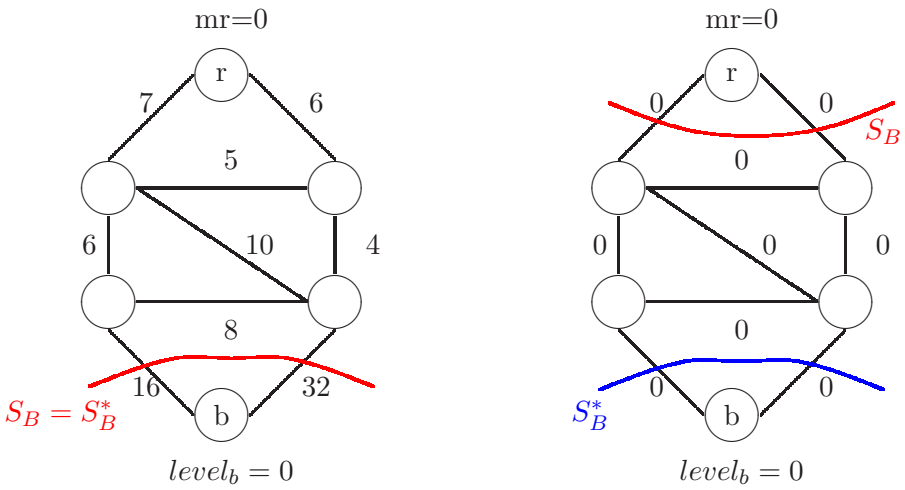

Figure 3: Examples of containment areas for $\mathcal{SP}$.

This construction allows us to follow a similar proof as in case 1 above (note that any process u which satisfies $\mu(u,r) \prec \Upsilon'$ will be disturb infinitely often, in particular at least p_{c+1} and p_{c+2} which contradicts the $(t,c,1)$-strong stabilization of $\mathcal{P}$).

In any case, we show that there exists a system which contradicts the $(t,c,1)$-strong stabilization of $\mathcal{P}$ that ends the proof. □

3.2 Topology Aware Strong Stabilization

First, we generalize the set S_B^* previously defined for the $\mathcal{BFS}$ metric in [6] to any maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$.

$$S_B^* = \left\{ v \in V \setminus B \,\middle|\, \mu(v,r) \prec \underset{b \in B}{max_{\prec}} \{\mu(v,b)\} \right\}$$

Intuitively, S_B^* gathers the set of corrects processes that are strictly closer (according to $\mathcal{M}$) to a Byzantine process than the root. Figures from 3 to 5 provide some examples of containment areas with respect to several maximizable metrics and compare it to S_B, the optimal containment area for TA strict stabilization.

Note that we assume for the sake of clarity that $V \setminus S_B^*$ induces a connected subsystem. If it is not the case, then S_B^* is extended to include all processes belonging to connected subsystems of $V \setminus S_B^*$ that not include r.

Now, we can state our generalization of Theorem 6.

Theorem 9 *Given a maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$, even under the central daemon, there exists no $(t, A_B^*, 1)$-TA-strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{M}$ where $A_B^* \subsetneq S_B^*$ and t is a given finite integer.*

Proof Let $\mathcal{M} = (M, W, mr, met, \prec)$ be a maximizable metric and $\mathcal{P}$ be a $(t, A_B^*, 1)$-TA-strongly stabilizing protocol for maximum metric spanning tree construction protocol with respect to $\mathcal{M}$ where $A_B^* \subsetneq S_B^*$ and t is a finite integer. We must distinguish the following cases:

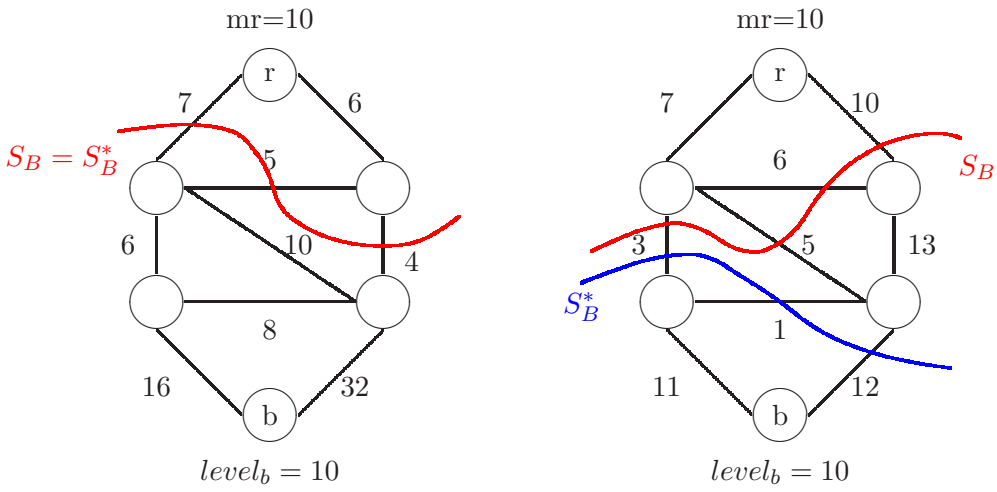

Figure 4: Examples of containment areas for $\mathcal{F}$.

Case 1: $|M| = 1$.

Denote by m the metric value such that $M = \{m\}$. For any system and for any process v, we have $\mu(v, r) = \underset{b \in B}{min_{\prec}}\{\mu(v, b)\} = m$. Consequently, $S_B^* = \emptyset$ for any system. Then, it is absurd to have $A_B^* \subsetneq S_B^*$.

Case 2: $|M| \geq 2$.

By definition of a bounded metric, we can deduce that there exists $m \in M$ and $w \in W$ such that $m = met(mr, w) \prec mr$. Then, we must distinguish the following cases:

Case 2.1: m is a fixed point of $\mathcal{M}$.

Let S be a system such that any edge incident to the root or a Byzantine process has a weight equals to w. Then, we can deduce that we have: $m = \underset{b \in B}{max_{\prec}}\{\mu(r, b)\} \prec \mu(r, r) = mr$ and for any correct process $v \neq r$, $\mu(v, r) = \underset{b \in B}{max_{\prec}}\{\mu(v, b)\} = m$. Hence, $S_B^* = \emptyset$ for any such system. Then, it is absurd to have $A_B^* \subsetneq S_B^*$.

Case 2.2: m is not a fixed point of $\mathcal{M}$.

This implies that there exists $w' \in W$ such that: $met(m, w') \prec m$ (remember that $\mathcal{M}$ is bounded). Consider the following system: $V = \{r, u, u', v, v', b\}$, $E = \{\{r, u\}, \{r, u'\}, \{u, v\}, \{u', v'\}, \{v, b\}, \{v', b\}\}$, $w_{r,u} = w_{r,u'} = w_{v,b} = w_{v',b} = w$, and $w_{u,v} = w_{u',v'} = w'$ (b is a Byzantine process). We can see that $S_B^* = \{v, v'\}$. Since $A_B^* \subsetneq S_B$, we have: $v \notin A_B^*$ or $v' \notin A_B^*$. Consider now the following configuration ρ_0: $prnt_r = prnt_b = \perp$, $level_r = level_b = mr$, $dist_r = dist_b = 0$ and $prnt$, $level$, and $dist$ variables of other processes are arbitrary (see Figure 6, other variables may have arbitrary values but other variables of b are identical to those of r).

Assume now that b takes exactly the same actions as r (if any) immediately after r (note that $r \notin A_B^*$ and hence $prnt_r = \perp$, $level_r = mr$, and $dist_r = 0$ still hold by closure and then $prnt_b = \perp$, $level_b = mr$, and $dist_r = 0$ still hold too). Then, by symmetry of the execution and by convergence of $\mathcal{P}$ to $spec$, we can deduce

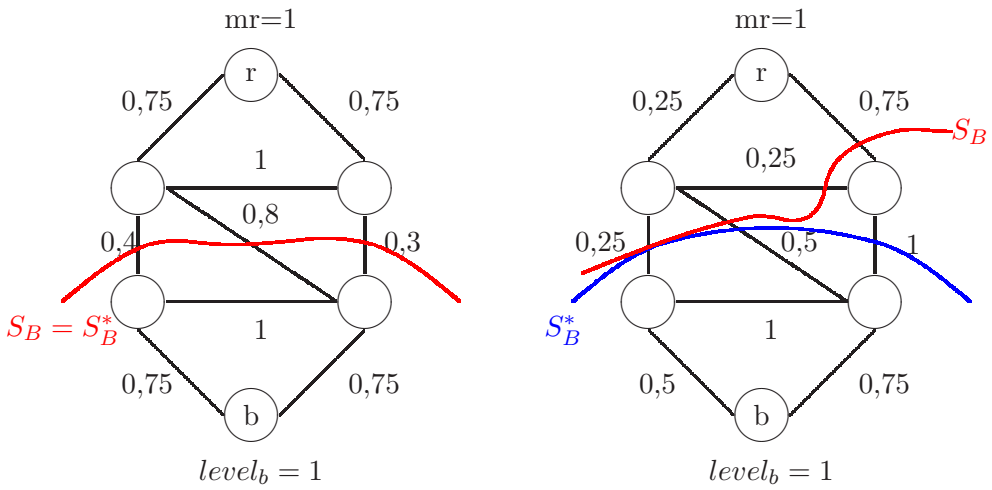

Figure 5: Examples of containment areas for $\mathcal{R}$.

that the system reaches in a finite time a configuration ρ_1 (see Figure 6) in which: $prnt_r = prnt_b = \perp$, $prnt_u = prnt_{u'} = r$, $prnt_v = prnt_{v'} = b$, $level_r = level_b = mr$, $level_u = level_{u'} = level_v = level_{v'} = m$, and $\forall v \in V, dist_v = legal\_dist_{prnt_v}$ (because this configuration is the only one in which all correct process v satisfies $spec(v)$ when $prnt_r = prnt_b = \perp$ and $level_r = level_b = mr$ since $met(m, w') \prec m$). Note that ρ_1 is A_B^*-legitimate for $spec$ and A_B^*-stable (whatever A_B^* is).

Assume now that b behaves as a correct processor with respect to $\mathcal{P}$. Then, by convergence of $\mathcal{P}$ in a fault-free system starting from ρ_1 which is not legitimate (remember that a TA-strongly stabilizing algorithm is a special case of self-stabilizing algorithm), we can deduce that the system reach in a finite time a configuration ρ_2 (see Figure 6) in which: $prnt_r = \perp$, $prnt_u = prnt_{u'} = r$, $prnt_v = u$, $prnt_{v'} = u'$, $prnt_b = v$ (or $prnt_b = v'$), $level_r = mr$, $level_u = level_{u'} = m$ $level_v = level_{v'} = met(m, w') = m'$, $level_b = met(m', w) = m''$, and $\forall v \in V, dist_v = legal\_dist_{prnt_v}$. Note that processes v and v' modify their O-variables in the portion of execution between ρ_1 and ρ_2 and that ρ_2 is A_B^*-legitimate for $spec$ and A_B^*-stable (whatever A_B^* is). Consequently, this portion of execution contains at least one A_B^*-TA-disruption (whatever A_B^* is).

Assume now that the Byzantine process b takes the following state: $prnt_b = \perp$ and $level_b = mr$. This step brings the system into configuration ρ_3 (see Figure 6). From this configuration, we can repeat the execution we constructed from ρ_0. By the same token, we obtain an execution of $\mathcal{P}$ which contains c-legitimate and c-stable configurations (see ρ_1) and an infinite number of A_B^*-TA-disruption (whatever A_B^* is) which contradicts the $(t, A_B^*, 1)$-TA-strong stabilization of $\mathcal{P}$.

□

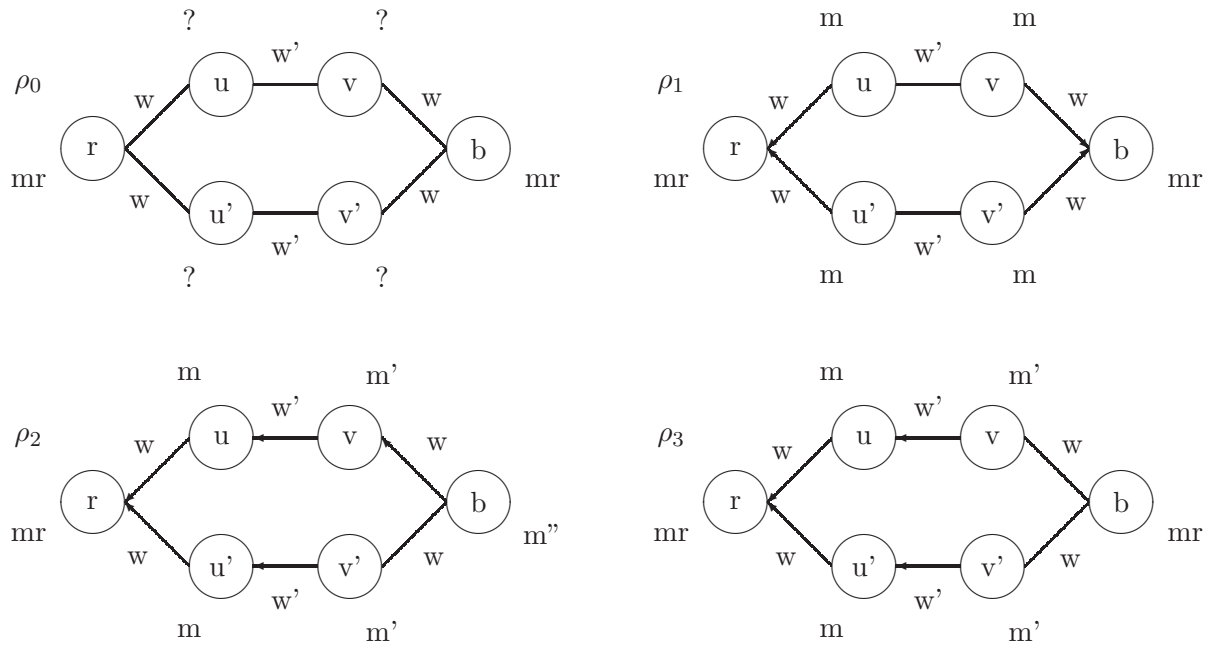

Figure 6: Configurations used in proof of Theorem 9.

4 Topology-Aware Strongly Stabilizing Protocol

The goal of this section is to provide a $(t, S_B^*, n-1)$-TA strongly stabilizing protocol in order to match the lower bound on containment area provided by the Theorem 9. If we focus on the protocol provided by [5] (which is $(S_B, n-1)$-TA strictly stabilizing), we can prove that this protocol does not satisfy our constraints since we have the following result.

Theorem 10 *Given a maximizable metric $\mathcal{M} = (M, W, mr, met, \prec)$, the protocol of [5] is not a $(t, S_B^*, 2)$-TA strongly stabilizing protocol for maximum metric spanning tree construction with respect to $\mathcal{M}$ where t is a given finite integer.*

Proof To prove this result, it is sufficient to construct an execution of the protocol of [5] for a given metric $\mathcal{M}$ which contains an infinite number of S_B^*-TA disruptions with two Byzantine processes.

Consider the shortest path metric $\mathcal{SP}$ defined above and the weighted system defined by Figure 7 (r denotes the root and b_1 and b_2 are two Byzantine processes). We recall that the protocol of [5] uses an upper bound D on the length of any path of the tree and that the protocol is built in such a way that a process cannot choose as parent a neighbor with a *dist* variable greater or equals to $D-1$. Here, we assume that $D = 10$.

If we consider the initial configuration ρ_1 defined by Figure 8, we can state that processes p_2 and p_3 cannot modify their state as long as b_1 remains in its state. Moreover, r and p_1 are never enabled by the protocol. In this way, it is possible to construct the following portion of execution e_1: b_2 modifies its level variable to 1. Then, p_5 and p_4 update their level variable to obtain configuration ρ_2 of Figure 8. Note that e_1 contains a S_B^*-TA disruption since p_4 modified one of its O-variables (namely, level) and $p_4 \notin S_B^*$. From ρ_2, it is possible to construct the following portion of execution e_2: b_2 modifies its level variable to 0. Then, p_5 and p_4 update their level variable to obtain configuration ρ_1.

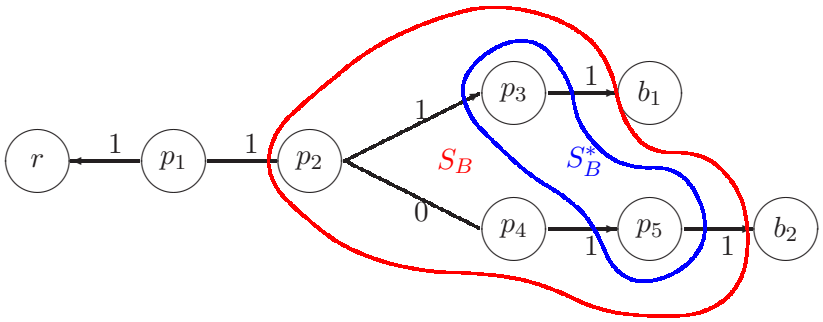

Figure 7: System used in proof of Theorem 10.

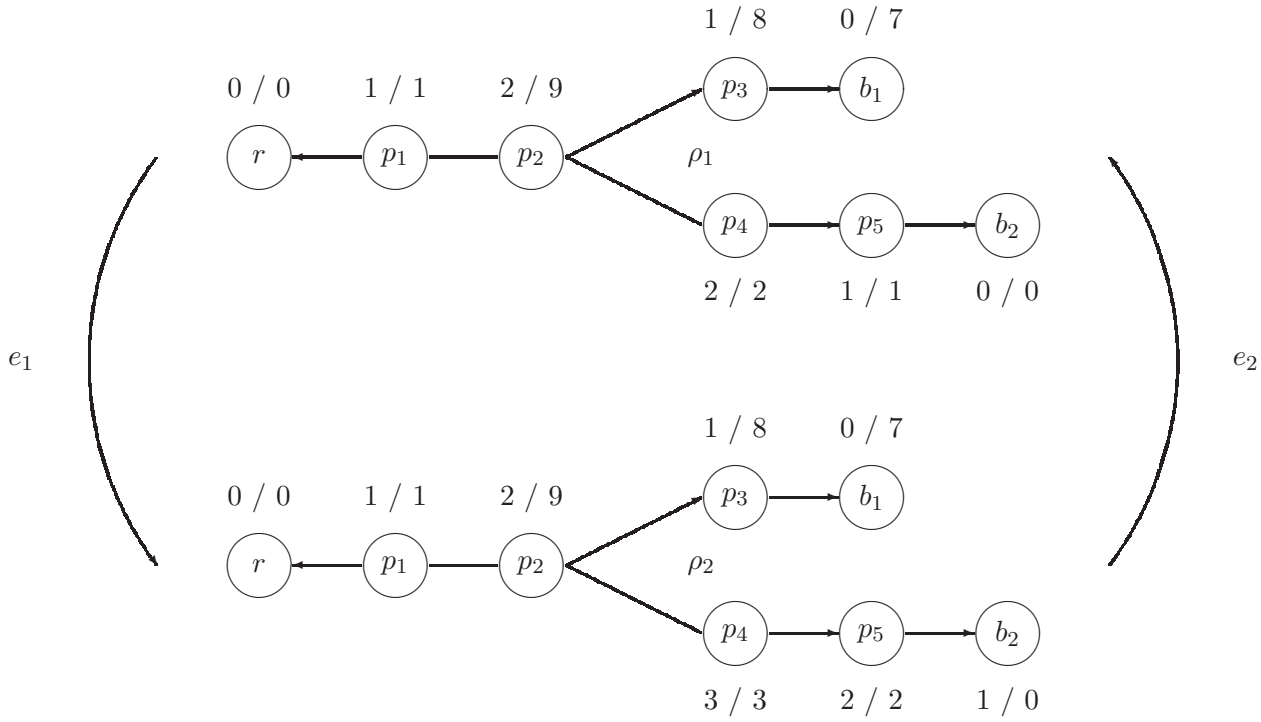

Figure 8: Configurations used in proof of Theorem 10 (for each process v, we use the notation $level_v$ / $dist_v$).

Consequently, it is possible to construct an infinite execution $e_1e_2e_1e_2\ldots$ starting from ρ_1 that contains an infinite number of S_B^*-TA disruptions with two Byzantine processes. This finishes the proof.

□

4.1 Presentation of the Protocol

In contrast of Theorem 10, we provide in this paper a new protocol which is $(t, S_B^*, n-1)$-TA strongly stabilizing for maximum metric spanning tree construction. Our protocol needs a supplementary assumption on the system. We introduce the following definition.

Definition 30 (Set of used metric values) *Given an assigned metric* $\mathcal{AM} = (M, W, met, mr, \prec, wf)$ *over a system* S*, the* set of used metric values *of* $\mathcal{AM}$ *is defined as* $M(S) = \{m \in M | \exists v \in V, (\mu(v, r) = m) \vee (\exists b \in B, \mu(v, b) = m)\}$.

We assume that we always have $|M(S)| \geq 2$ (the necessity of this assumption is explained below). Nevertheless, note that the contrary case ($|M(S)| = 1$) is possible if and only if the assigned metric is equivalent to $\mathcal{NC}$. As the protocol of [7] performs $(t, 0, n-1)$- strong stabilization with a finite t for this metric, we can achieves the $(t, S_B^*, n-1)$-TA strong stabilization when $|M(S)| = 1$ (since this implies that $S_B^* = \emptyset$). In this way, this assumption does not weaken the possibility result.

Although the protocol of [5] is not TA strongly stabilizing (see Theorem 10), our protocol borrows fundamental strategy from it. In this protocol, any process try to maximize its *level* in the tree by choosing as its parent the neighbor that provide the best metric value. The key idea of this protocol is to use the distance variable (upper bounded by a given constant D) to detect and break cycles of process which has the same maximum metric. To achieve the TA strict stabilization, the protocol ensures a fair selection along the set of its neighbor with a round-robin order.

The possibility of infinite number of disruptions of the protocol of [5] mainly comes from the following fact: a Byzantine process can independently lie about its *level* and its *dist* variable. For example, a Byzantine process can provide a *level* equals to *mr* and a *dist* arbitrarily large. In this way, it may lead a correct process of $S_B \setminus S_B^*$ to have a *dist* variable equals to $D-1$ such that no other correct process can choose it as its parent (this rule is necessary to break cycle) but it cannot modify its state (this rule is only enabled when *dist* is equals to D). Then, this process may always prevent some of its neighbors to join a $\mathcal{M}$-path connected to the root and hence allow another Byzantine process to perform an infinite number of disruptions.

It is why we modified the management of the *dist* variable (note that others variables are managed exactly in the same way as in the protocol of [5]). In order to contain the effect of Byzantine process on *dist* variables, each process that has a *level* different from the one of its parent in the tree sets its *dist* variable to 0. In this way, a Byzantine process modifying its *dist* variable can only affect correct process that have the same *level*. Consequently, in the case where $|M(S)| \geq 2$, we are ensured that correct processes of $S_B \setminus S_B^*$ cannot keep a *dist* variable equals or greater than $D-1$ infinitely. Hence, a correct process of $S_B \setminus S_B^*$ cannot be disturbed infinitely often without joining a $\mathcal{M}$-path connected to the root.

We can see that the assumption $|M(S)| \geq 2$ is essential to perform the topology-aware strong stabilization. Indeed, in the case where $|M(S)| = 1$, Byzantine processes can play exactly the scenario described above (in this case, our protocol is equivalent to the one of [5]).

The second modification we bring to the protocol of [5] follows. When a process has an inconsistent *dist* variable with its parent, we allow it only to increase its *dist* variable. If the

process needs to decrease its *dist* variable (when it has a strictly greater distance than its parent), then the process must change its parent. This rule allows us to bound the maximal number of steps of any process between two modifications of its parent (a Byzantine process cannot lead a correct one to infinitely often increase and decrease its distance without modifying its pointer).

Our protocol is formally described in Algorithm 4.1.

algorithm 4.1 $\mathcal{SSMAX}$, TA strongly stabilizing protocol for maximum metric tree construction.

Data:

N_v: totally ordered set of neighbors of v.
D: upper bound of the number of processes in a simple path.

Variables:

$prnt_v \in \begin{cases} \{\perp\} \text{ if } v = r \\ N_v \text{ if } v \neq r \end{cases}$: pointer on the parent of v in the tree.
$level_v \in \{m \in M | m \preceq mr\}$: metric of the node.
$dist_v \in \{0, \ldots, D\}$: hop counter.

Functions:

For any subset $A \subseteq N_v$, $choose_v(A)$ returns the first element of A which is bigger than $prnt_v$ (in a round-robin fashion).

$current\_dist_v() = \begin{cases} 0 \text{ if } level_{prnt_v} \neq level_v \\ min(dist_{prnt_v} + 1, D) \text{ if } level_{prnt_v} = level_v \end{cases}$

Rules:

$(\boldsymbol{R_r})$ :: $(v = r) \wedge ((level_v \neq mr) \vee (dist_v \neq 0)) \longrightarrow level_v := mr;\ dist_v := 0$

$(\boldsymbol{R_1})$ :: $(v \neq r) \wedge (prnt_v \in N_v) \wedge ((dist_v < current\_dist_v()) \vee (level_v \neq met(level_{prnt_v}, w_{v,prnt_v})))$
$\longrightarrow level_v := met(level_{prnt_v}, w_{v,prnt_v});\ dist_v := current\_dist_v()$

$(\boldsymbol{R_2})$ :: $(v \neq r) \wedge ((dist_v = D) \vee (dist_v > current\_dist_v())) \wedge (\exists u \in N_v, dist_u < D - 1)$
$\longrightarrow prnt_v := choose_v(\{u \in N_v | dist_v < D - 1\});\ level_v := met(level_{prnt_v}, w_{v,prnt_v});\ dist_v := current\_dist_v()$

$(\boldsymbol{R_3})$ :: $(v \neq r) \wedge (\exists u \in N_v, (dist_u < D - 1) \wedge (level_v \prec met(level_u, w_{u,v})))$
$\longrightarrow prnt_v := choose_v\left(\left\{u \in N_v \middle| (level_u < D - 1) \wedge (met(level_u, w_{u,v}) = \max_{\prec_{q \in N_v / level_q < D-1}} \{met(level_q, w_{q,v})\})\right\}\right);$
$level_v := met(level_{prnt_v}, w_{prnt_v,v});\ dist_v := current\_dist_v()$

4.2 Proof of the $(S_B, n-1)$-TA Strict Stabilization for *spec*

This proof is similar to the one of [5] but we must modify it to take in account modifications of the protocol. In [5], we proved the following useful property about maximizable metrics.

Lemma 1 *For any process $v \in V$, we have:*

$$\forall u \in N_v, met\left(\max_{\prec_{p \in B \cup \{r\}}} \{\mu(u,p)\}, w_{u,v}\right) \preceq \max_{\prec_{p \in B \cup \{r\}}} \{\mu(v,p)\}$$

Given a configuration $\rho \in C$ and a metric value $m \in M$, let us define the following predicate:

$$IM_m(\rho) \equiv \forall v \in V, level_v \preceq max_{\prec} \left\{m, \max_{\prec_{u \in B \cup \{r\}}} \{\mu(v,u)\}\right\}$$

Lemma 2 *For any metric value $m \in M$, the predicate IM_m is closed by actions of $\mathcal{SSMAX}$.*

Proof Let m be a metric value ($m \in M$). Let $\rho \in C$ be a configuration such that $IM_m(\rho) = true$ and $\rho' \in C$ be a configuration such that $\rho \overset{R}{\mapsto} \rho'$ is a step of $\mathcal{SSMAX}$.

If the root process $r \in R$ (respectively a Byzantine process $b \in R$), then we have $level_r = mr$ (respectively $level_b \preceq mr$) in ρ' by construction of $(\boldsymbol{R_r})$ (respectively by definition of $level_b$). Hence, $level_r \preceq max_\prec \left\{ m,\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(r,u)\} \right\} = mr$ (respectively $level_b \preceq max_\prec \left\{ m,\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(b,u)\} \right\} = mr$).

If a correct process $v \in R$ with $v \neq r$, then there exists a neighbor p of v such that $level_p \preceq max_\prec \left\{ m,\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(p,u)\} \right\}$ in ρ (since $IM_m(\rho) = true$) and $prnt_v = p$ and $level_v = met(level_p,\ w_{v,p})$ in ρ' (since v is activated during this step).

If we apply the Lemma 1 to met and to neighbor p, we obtain the following property:

$$met\left(\underset{u \in B \cup \{r\}}{max_\prec} \{\mu(p,u)\}, w_{v,p} \right) \preceq \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(v,u)\}$$

Consequently, we obtain that, in ρ':

$$\begin{array}{rcll}
level_v & = & met(level_p, w_{v,p}) & \\
& \preceq & met\left(max_\prec \left\{ m,\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(p,u)\} \right\}, w_{v,p} \right) & \text{by monotonicity of } \mathcal{M} \\
& \preceq & max_\prec \left\{ met(m, w_{v,p}), met\left(\underset{u \in B \cup \{r\}}{max_\prec} \{\mu(p,u)\}, w_{v,p} \right) \right\} & \\
& \preceq & max_\prec \left\{ m,\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(v,u)\} \right\} & \text{since } met(m, w_{v,p}) \preceq m
\end{array}$$

We can deduce that $IM_m(\rho') = true$, that concludes the proof. □

Given an assigned metric to a system G, we can observe that the set of metrics value M is finite and that we can label elements of M by $m_0 = mr, m_1, \ldots, m_k$ in a way such that $\forall i \in \{0, \ldots, k-1\}, m_{i+1} \prec m_i$.

We introduce the following notations:

$$\begin{array}{lrcl}
\forall m_i \in M, & P_{m_i} & = & \left\{ v \in V \setminus S_B \middle| \mu(v,r) = m_i \right\} \\
\forall m_i \in M, & V_{m_i} & = & \bigcup_{j=0}^{i} P_{m_j} \\
\forall m_i \in M, & I_{m_i} & = & \left\{ v \in V \middle|\ \underset{u \in B \cup \{r\}}{max_\prec} \{\mu(v,u)\} \prec m_i \right\} \\
\forall m_i \in M, & \mathcal{LC}_{m_i} & = & \left\{ \rho \in \mathcal{C} \middle| (\forall v \in V_{m_i}, spec(v)) \wedge (IM_{m_i}(\rho)) \right\} \\
& \mathcal{LC} & = & \mathcal{LC}_{m_k}
\end{array}$$

Lemma 3 *For any $m_i \in M$, the set $\mathcal{LC}_{m_i}$ is closed by actions of $\mathcal{SSMAX}$.*

Proof Let m_i be a metric value from M and ρ be a configuration of $\mathcal{LC}_{m_i}$. By construction, any process $v \in V_{m_i}$ satisfies $spec(v)$ in ρ.

In particular, the root process satisfies: $prnt_r = \perp$, $level_r = mr$, and $dist_r = 0$. By construction of $\mathcal{SSMAX}$, r is not enabled and then never modifies its O-variables (since the guard of the rule of r does not involve the state of its neighbors).

In the same way, any process $v \in V_{m_i}$ satisfies: $prnt_v \in N_v$, $level_v = met(level_{prnt_v}, w_{prnt_v,v})$, $dist_v = legal\_dist_{prnt_v}$, and $level_v = \underset{u \in N_v}{max_{\prec}}\{met(level_u, w_{u,v})\}$. Note that, as $v \in V_{m_i}$ and $spec(v)$ holds in ρ, we have: $level_v = \mu(v,r) = \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(v,p)\}$ and $dist_v \leq D-1$ by construction of D. Hence, process v is not enabled in ρ.

Assume that there exists a process $v \in V_{m_i}$ that takes a step $\rho' \overset{R}{\mapsto} \rho''$ in an execution starting from ρ (without loss of generality, assume that v is the first process of $v \in V_{m_i}$ that takes a step in this execution). Then, we know that $v \neq r$. This activation implies that a neighbor $u \notin V_{m_i}$ (since v is the first process of V_{m_i} to take a step) of v modified its $level_u$ variable to a metric value $m \in M$ such that $level_v \prec met(m, w_{u,v})$ in ρ' (note that O-variables of v and $prnt_v$ remain consistent since v is the first process to take a step in this execution).

Hence, we have $level_v = \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(v,p)\} = \mu(v,r)$ (since $spec(v)$ holds), $level_v \prec met(m, w_{u,v})$ (since u causes an action of v), and $m_i \preceq level_v$ (since $v \in V_{m_i}$ and $level_v = \mu(v,r)$). Moreover, the closure of IM_{m_i} (established in Lemma 2) ensures us that $m = level_u \preceq max_{\prec}\left\{m_i,\ \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}\right\}$. Let us study the two following cases:

Case 1: $max_{\prec}\left\{m_i,\ \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}\right\} = m_i$.

We have then $m \preceq m_i$. As the boundedness of $\mathcal{M}$ ensures that $met(m, w_{u,v}) \preceq m$, we can conclude that $level_v \prec met(m, w_{u,v}) \preceq m \preceq m_i \preceq level_v$, that is absurd.

Case 2: $max_{\prec}\left\{m_i,\ \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}\right\} = \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}$.

We have then $m \preceq \underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}$. By monotonicity of $\mathcal{M}$, we can deduce that $met(m, w_{u,v}) \preceq met(\underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}, w_{u,v})$. Consequently, we obtain that $\underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(v,p)\} \prec met(\underset{p \in B \cup \{r\}}{max_{\prec}}\{\mu(u,p)\}, w_{u,v})$. This is contradictory with the result of Lemma 1.

In conclusion, any process $v \in V_{m_i}$ takes no step in any execution starting from ρ and then always satisfies $spec(v)$. Then, the closure of IM_B (established in Lemma 2) concludes the proof. □

Lemma 4 *Any configuration of $\mathcal{LC}$ is $(S_B, n-1)$-TA contained for spec.*

Proof This is a direct application of the Lemma 3 to $\mathcal{LC} = \mathcal{LC}_{m_k}$. □

Lemma 5 *Starting from any configuration of $\mathcal{C}$, any execution of $\mathcal{SSMAX}$ reaches in a finite time a configuration of $\mathcal{LC}_{mr}$.*

Proof Let ρ be an arbitrary configuration. Then, it is obvious that $IM_{mr}(\rho)$ is satisfied. By closure of IM_{mr} (proved in Lemma 2), we know that IM_{mr} remains satisfied in any execution starting from ρ.

If r does not satisfy $spec(r)$ in ρ, then r is continuously enabled. Since the scheduling is strongly fair, r is activated in a finite time and then r satisfies $spec(r)$ in a finite time. Denote by ρ' the first configuration in which $spec(r)$ holds. Note that r takes no step in any execution starting from ρ'.

The boundedness of $\mathcal{M}$ implies that P_{mr} induces a connected subsystem. If $P_{mr} = \{r\}$, then we proved that $\rho' \in \mathcal{LC}_{mr}$ and we have the result.

Otherwise, observe that, for any configuration of an execution starting from ρ', if all processes of P_{mr} are not enabled, then all processes v of P_{mr} satisfy $spec(v)$. Assume now that there exists an execution e starting from ρ' in which some processes of P_{mr} take infinitely many steps. By construction, at least one of these processes (note it v) has a neighbor u which takes only a finite number of steps in e (recall that P_{mr} induces a connected subsystem and that r takes no step in e). After u takes its last step of e, we can observe that $level_u = mr$ and $dist_u < D - 1$ (otherwise, u is activated in a finite time that contradicts its construction).

As v can execute consequently $\boldsymbol{(R_1)}$ only a finite number of times (since the incrementation of $dist_v$ is bounded by D), we can deduce that v executes $\boldsymbol{(R_2)}$ or $\boldsymbol{(R_3)}$ infinitely often. In both cases, u belongs to the set which is the parameter of function *choose*. By the fairness of this function, we can deduce that $prnt_v = u$ in a finite time in e. Then, the construction of u implies that v is never enabled in the sequel of e. This is contradictory with the construction of e.

Consequently, any execution starting from ρ' reaches in a finite time a configuration such that all processes of P_{mr} are not enabled. We can deduce that this configuration belongs to $\mathcal{LC}_{mr}$, that ends the proof. □

Lemma 6 *For any $m_i \in M$ and for any configuration $\rho \in \mathcal{LC}_{m_i}$, any execution of $\mathcal{SSMAX}$ starting from ρ reaches in a finite time a configuration such that:*

$$\forall v \in I_{m_i}, level_v = m_i \Rightarrow dist_v = D$$

Proof Let m_i be an arbitrary metric value of M and ρ_0 be an arbitrary configuration of $\mathcal{LC}_{m_i}$. Let $e = \rho_0, \rho_1, \ldots$ be an execution starting from ρ_0.

Note that ρ_0 satisfies IM_{m_i} by construction. Hence, we have $\forall v \in I_{m_i}, level_v \preceq m_i$. The closure of IM_{m_i} (proved in Lemma 2) ensures us that this property is satisfied in any configuration of e.

If any process $v \in I_{m_i}$ satisfies $level_v \prec m_i$ in ρ_0, then the result is obvious. Otherwise, we define the following variant function. For any configuration ρ_j of e, we denote by A_j the set of processes v of I_{m_i} such that $level_v = m_i$ in ρ_j. Then, we define $f(\rho_j) = \min\limits_{v \in A_j}\{dist_v\}$. We will prove the result by showing that there exists an integer k such that $f(\rho_k) = D$.

First, if a process v joins A_j (that is, $v \notin A_{j-1}$ but $v \in A_j$), then it takes a distance value greater or equals to $f(\rho_{j-1}) + 1$ by construction of the protocol. We can deduce that any process that joins A_j does not decrease f. Moreover, the construction of the protocol implies that a process v such that $v \in A_j$ and $v \in A_{j+1}$ can not decrease its distance value in the step $\rho_j \mapsto \rho_{j+1}$.

Then, consider for a given configuration ρ_j a process $v \in A_j$ such that $dist_v = f(\rho_j) < D$. We claim that v is enabled in ρ_j and that the execution of the enabled rule either increases strictly $dist_v$ or removes v from A_{j+1}. We distinguish the following cases:

Case 1: $level_v = met(level_{prnt_v}, w_{v,prnt_v})$

The fact that $v \in I_{m_i}$, the boundedness of $\mathcal{M}$ and the closure of IM_{m_i} imply that $prnt_v \in A_j$ (and, hence that $level_{prnt_v} = m_i$). Then, by construction of $f(\rho_j)$, we know that $dist_{prnt_v} \geq f(\rho_j) = dist_v$. Hence, we have $dist_v < dist_{prnt_v} + 1$ in ρ_j. Then, v is enabled by $\boldsymbol{(R_1)}$ in ρ_j and $dist_v$ increases of at least 1 during the step $\rho_j \mapsto \rho_{j+1}$ if this rule is executed.

Case 2: $level_v \neq met(level_{prnt_v}, w_{v,prnt_v})$

Assume that v is activated by $(\boldsymbol{R_2})$ or $(\boldsymbol{R_3})$ during the step $\rho_j \mapsto \rho_{j+1}$. If v does not belong to A_{j+1} (if $level_v \neq m_i$ in ρ_{j+1}), the claim is satisfied. In the contrary case (v belongs to A_{j+1}), we know that $level_v = m_i$ in ρ_{j+1}. The boundedness of $\mathcal{M}$ and the closure of IM_{m_i} imply that $level_{prnt_v} = m_i$ in ρ_{j+1}. We can conclude that $dist_v$ increases of at least 1 during the step $\rho_j \mapsto \rho_{j+1}$ since the new parent of v has a distance greater than $f(\rho_j)$ by construction of A_{j+1}.

Otherwise, we know that the rule $(\boldsymbol{R_1})$ is enabled for v in ρ_j. If this rule is executed during the step $\rho_j \mapsto \rho_{j+1}$, one of the two following sub cases appears.

Case 2.1: $met(level_{prnt_v}, w_{v,prnt_v}) \prec m_i$ in ρ_j.

Then, v does not belong to A_{j+1} by definition.

Case 2.2: $met(level_{prnt_v}, w_{v,prnt_v}) = m_i$ in ρ_j.

Remind that the closure of IM_{m_i} implies then that $level_{prnt_v} = m_i$. By construction of $f(\rho_j)$, we have $dist_{prnt_v} \geq f(\rho_j)$ in ρ_j. Then, we can see that $dist_v$ increases of at least 1 during the step $\rho_j \mapsto \rho_{j+1}$.

In all cases, v is enabled (at least by $(\boldsymbol{R_1})$) in ρ_j and the execution of the enabled rule either increases strictly $dist_v$ or removes v from A_{j+1}.

As I_{m_i} is finite and the scheduling is strongly fair, we can deduce that f increases in a finite time in any execution starting from ρ_j. By repeating the argument at most D times, we can deduce that e contains a configuration ρ_k such that $f(\rho_k) = D$, that shows the result. □

Lemma 7 *For any $m_i \in M$ and for any configuration $\rho \in \mathcal{LC}_{m_i}$ such that $\forall v \in I_{m_i}, level_v = m_i \Rightarrow dist_v = D$, any execution of $\mathcal{SSMAX}$ starting from ρ reaches in a finite time a configuration such that:*

$$\forall v \in I_{m_i}, level_v \prec m_i$$

Proof Let $m_i \in M$ be an arbitrary metric value and ρ_0 be a configuration of $\mathcal{LC}_{m_i}$ such that $\forall v \in I_{m_i}, level_v = m_i \Rightarrow dist_v = D$. Let $e = \rho_0, \rho_1, \ldots$ be an arbitrary execution starting from ρ_0.

For any configuration ρ_j of e, let us denote $E_{\rho_j} = \{v \in I_{m_i} | level_v = m_i\}$. By the closure of IM_{m_i} (which holds by definition in ρ_0) established in Lemma 2, we obtain the result if there exists a configuration ρ_j of e such that $E_{\rho_j} = \emptyset$.

If there exist some processes $v \in I_{m_i} \backslash E_{\rho_0}$ (and hence $level_v \prec m_i$) such that $prnt_v \in E_{\rho_0}$ and $met(level_{prnt_v}, w_{v,prnt_v}) = m_i$ in ρ_0, then we can observe that these processes are continuously enabled by $(\boldsymbol{R_1})$. As the scheduling is strongly fair, v activates this rule in a finite time and then, $level_v = m_i$ and $dist_v = D$. In other words, v joins E_{ρ_l} for a given integer l. We can conclude that there exists an integer k such that the following property $(\boldsymbol{P})$ holds: for any $v \in I_{m_i} \setminus E_{\rho_0}$, either $prnt_v \notin E_{\rho_k}$ or $met(level_{prnt_v}, w_{v,prnt_v}) \prec m_i$.

Then, we prove that, for any integer $j \geq k$, we have $E_{\rho_{j+1}} \subseteq E_{\rho_j}$. For the sake of contradiction, assume that there exists an integer $j \geq k$ and a process $v \in I_{m_i}$ such that $v \in E_{\rho_{j+1}}$ and $v \notin E_{\rho_j}$. Without loss of generality, assume that j is the smallest integer which performs these properties. Let us study the following cases:

Case 1: v activates $(\boldsymbol{R_1})$ during the step $\rho_j \mapsto \rho_{j+1}$.

Note that the property $(\boldsymbol{P})$ still holds in ρ_j by the construction of j. Hence, we know that $prnt_v \notin E_{\rho_j}$ in ρ_j. But in this case, we have: $level_{prnt_v} \prec m_i$. The boundedness of $\mathcal{M}$ implies that $level_v = met(level_{prnt_v}, w_{v,prnt_v}) \prec m_i$ in ρ_{j+1} that contradicts the fact that $v \in E_{\rho_{j+1}}$.

Case 2: v activates either $(\boldsymbol{R_2})$ or $(\boldsymbol{R_3})$ during the step $\rho_j \mapsto \rho_{j+1}$.
That implies v chooses a new parent which has a distance smaller than $D-1$ in ρ_j. This implies that this new parent does not belongs to E_{ρ_j}. Then, we have $level_{prnt_v} \prec m_i$. The boundedness of $\mathcal{M}$ implies that $level_v = met(level_{prnt_v}, w_{v,prnt_v}) \prec m_i$ in ρ_{j+1} that contradicts the fact that $v \in E_{\rho_{j+1}}$.

In the two cases, our claim is satisfied. In other words, there exists a point of the execution (namely ρ_k) afterwards the set E cannot grow (this implies that, if a process leaves the set E, it is a definitive leaving).

Assume now that there exists a step $\rho_j \mapsto \rho_{j+1}$ (with $j \geq k$) such that a process $v \in E_{\rho_j}$ is activated. Observe that the closure of IM_{m_i} implies that v can not be activated by the rule $(\boldsymbol{R_3})$. If v activates $(\boldsymbol{R_1})$ during this step, then v modifies its level during this step (otherwise, we have a contradiction with the fact that $level_{prnt_v} = m_i \Rightarrow dist_v = D$). The closure of IM_{m_i} implies that v leaves the set E during this step. If v activates $(\boldsymbol{R_2})$ during this step, then v chooses a new parent which has a distance smaller than $D-1$ in ρ_j. This implies that this new parent does not belongs to E_{ρ_j}. Then, we have $level_{prnt_v} \prec m_i$. The boundedness of $\mathcal{M}$ implies that $level_v \prec m_i$ in ρ_{j+1}. In other words, if a process of E_{ρ_j} is activated during the step $\rho_j \mapsto \rho_{j+1}$, then it satisfies $v \notin E_{\rho_{j+1}}$.

Finally, observe that the construction of the protocol and the construction of the bound D ensures us that any process $v \in I_{m_i}$ such that $dist_v = D$ is activated in a finite time. In conclusion, we obtain that there exists an integer j such that $E_{\rho_j} = \emptyset$, that implies the result. □

Lemma 8 *For any $m_i \in M$ and for any configuration $\rho \in \mathcal{LC}_{m_i}$, any execution of $\mathcal{SSMAX}$ starting from ρ reaches in a finite time a configuration ρ' such that $IM_{m_{i+1}}$ holds.*

Proof This result is a direct consequence of Lemmas 6 and 7. □

Lemma 9 *For any $m_i \in M$ and for any configuration $\rho \in \mathcal{LC}_{m_i}$, any execution of $\mathcal{SSMAX}$ starting from ρ reaches in a finite time a configuration of $\mathcal{LC}_{m_{i+1}}$.*

Proof Let m_i be a metric value of M and ρ be an arbitrary configuration of $\mathcal{LC}_{m_i}$. We know by Lemma 8 that any execution starting from ρ reaches in a finite time a configuration ρ' such that $IM_{m_{i+1}}$ holds. By closure of $IM_{m_{i+1}}$ and of $\mathcal{LC}_{m_i}$ (established respectively in Lemma 2 and 3), we know that any configuration of any execution starting from ρ' belongs to $\mathcal{LC}_{m_i}$ and satisfies $IM_{m_{i+1}}$.

We know that $V_{m_i} \neq \emptyset$ since $r \in V_{m_i}$ for any $i \geq 0$. Remind that $V_{m_{i+1}}$ is connected by the boundedness of $\mathcal{M}$. Then, we know that there exists at least one process p of $P_{m_{i+1}}$ which has a neighbor q in V_{m_i} such that $\mu(p,r) = met(\mu(q,r), w_{p,q})$. Moreover, Lemma 3 ensures us that any process of V_{m_i} takes no step in any executions starting from ρ'.

Observe that, for any configuration of an execution starting from ρ', if any process of $P_{m_{i+1}}$ is not enabled, then all processes v of $P_{m_{i+1}}$ satisfy $spec(v)$. Assume now that there exists an execution e starting from ρ' in which some processes of $P_{m_{i+1}}$ take infinitely many steps. By construction, at least one of these processes (note it v) has a neighbor u such that $\mu(v,r) = met(\mu(u,r), w_{v,u})$ which takes only a finite number of steps in e (recall the construction of p). After u takes its last step of e, we can observe that $level_u = \mu(u,r)$ and $dist_u < D-1$ (otherwise, u is activated in a finite time that contradicts its construction).

As v can execute consequently $(\boldsymbol{R_1})$ only a finite number of times (since the incrementation of $dist_v$ is bounded by D), we can deduce that v executes $(\boldsymbol{R_2})$ or $(\boldsymbol{R_3})$ infinitely often. In both cases, u belongs to the set which is the parameter of function *choose* (remind that $IM_{m_{i+1}}$

is satisfied and that u has the better possible metric among v's neighbors). By the construction of this function, we can deduce that $prnt_v = u$ in a finite time in e. Then, the construction of u implies that v is never enabled in the sequel of e. This is contradictory with the construction of e.

Consequently, any execution starting from ρ' reaches in a finite time a configuration such that all processes of $P_{m_{i+1}}$ are not enabled. We can deduce that this configuration belongs to $\mathcal{LC}_{m_{i+1}}$, that ends the proof. □

Lemma 10 *Starting from any configuration, any execution of $\mathcal{SSMAX}$ reaches a configuration of $\mathcal{LC}$ in a finite time.*

Proof Let ρ be an arbitrary configuration. We know by Lemma 5 that any execution starting from ρ reaches in a finite time a configuration of $\mathcal{LC}_{mr} = \mathcal{LC}_{m_0}$. Then, we can apply at most k times the result of Lemma 9 to obtain that any execution starting from ρ reaches in a finite time a configuration of $\mathcal{LC}_{m_k} = \mathcal{LC}$, that proves the result. □

Theorem 11 *$\mathcal{SSMAX}$ is a $(S_B, n-1)$-TA-strictly stabilizing protocol for spec.*

Proof This result is a direct consequence of Lemmas 4 and 10. □

4.3 Proof of the $(t, S_B^*, n-1)$-TA Strong Stabilization for *spec*

Let be $E_B = S_B \setminus S_B^*$ (*i.e.* E_B is the set of process v such that $\mu(v,r) = \underset{b \in B}{max}\{\mu(v,b)\}$). Note that the subsytem induced by E_B may have several connected components. In the following, we use the following notations: $E_B = \{E_B^1, \ldots, E_B^\ell\}$ where each E_B^i ($i \in \{0, \ldots, \ell\}$) is a subset of E_B inducing a maximal connected component, $\delta(E_B^i)$ ($i \in \{0, \ldots, \ell\}$) is the diameter of the subsystem induced by E_B^i, and $\delta = \underset{i \in \{0,\ldots,\ell\}}{max}\{\delta(E_B^i)\}$. When a and b are two integers, we define the following function: $\Pi(a,b) = \frac{a^{b+1}-1}{a-1}$.

Lemma 11 *If ρ is a configuration of $\mathcal{LC}$, then any process $v \in E_B$ is activated at most $\Pi(k,\delta)\Delta D$ times in any execution starting from ρ.*

Proof Let ρ be a configuration of $\mathcal{LC}$ and e be an execution starting from ρ. Let p be a process of E_B^i ($i \in \{0, \ldots, \ell\}$) such that there exists a neighbor q which satisfies $q \in V \setminus S_B$ and $\mu(p,r) = met(\mu(q,r), w_{p,q})$ (such a process exists by construction of E_B^i). We are going to prove by induction on d the following property:

(P_d): if v is a process of E_B^i such that $d_{E_B^i}(p,v) = d$ (where $d_{E_B^i}$ denotes the distance in the subsystem induced by E_B^i), then v executes at most $\Pi(k,d)\Delta D$ actions in e.

Initialization: $d = 0$.

This implies that $v = p$. Then, by construction, there exists a neighbor q which satisfies $q \in V \setminus S_B$ and $\mu(p,r) = met(\mu(q,r), w_{p,q})$. As $\rho \in \mathcal{LC}$, Lemma 4 ensures us that $level_q = \mu(q,r)$ and $dist_q < D-1$ in any configuration of e. Then, the boundedness of $\mathcal{M}$ implies that q belongs to the set which is parameter to the macro *choose* at any execution of rules **(R_2)** or **(R_3)** by p. Consequently, p executes at most Δ times rules **(R_2)** and **(R_3)** in e before choosing q as its parent. Moreover, note that p can execute rule **(R_1)** at most D times between two consecutive executions of rules **(R_2)** and **(R_3)** (because **(R_1)** only increases $dist_p$ which is bounded by D). Consequently, p executes at most ΔD actions before choosing q as its parent.

By Lemma 4, we know that q takes no action in e. Once p chooses q as its parent, its state is consistent with the one of q (by construction of rules $(\boldsymbol{R_2})$ and $(\boldsymbol{R_3})$). Hence, p is never enabled after choosing q as its parent. Consequently, we obtain that p takes at most ΔD actions in e, that proves $(\boldsymbol{P_0})$.

Induction: $d > 0$ and $(\boldsymbol{P_{d-1}})$ is true.

Let v be a process of E_B^i such that $d_{E_B^i}(p, v) = d$. By construction, there exists a neighbor u of v which belongs to E_B^i such that $d_{E_B^i}(p, u) = d - 1$. By $(\boldsymbol{P_{d-1}})$, we know that u takes at most $\Pi(k, d-1)\Delta D$ actions in e. The k-boundedness of the daemon allows us to conclude that v takes at most $k \times \Pi(k, d-1)\Delta D$ actions before the last action of u. Then, a similar reasoning to the one of the initialization part allows us to say that v takes at most ΔD actions after the last action of u (note that the fact that $|M(S)| \geq 2$, the construction of D and the management of *dist* variables imply that $dist_u < D - 1$ after the last step of u). In conclusion, v takes at most $k \times \Pi(k, d-1)\Delta D + \Delta D = \Pi(k, d)\Delta D$ actions in e, that proves $(\boldsymbol{P_d})$.

As δ denotes the maximal diameter of connected components of the subsystem induced by E_B, then we know that $d_{E_B^i}(p, v) \leq \delta$ for any process v in E_B^i. For any process v of E_B, there exists $i \in \{0, \ldots, \ell\}$ such that $v \in E_B^i$. We can deduce that any process of E_B takes at most $\Pi(k, \delta)\Delta D$ actions in e, that implies the result. □

Lemma 12 *If ρ is a configuration of $\mathcal{LC}$ and v is a process such that $v \in E_B$, then for any execution e starting from ρ either*

1. *there exists a configuration ρ' of e such that $spec(v)$ is always satisfied after ρ', or*
2. *v is activated in e.*

Proof Let ρ be a configuration of $\mathcal{LC}$ and v be a process such that $v \in E_B$. By contradiction, assume that there exists an execution starting from ρ such that (i) $spec(v)$ is infinitely often false in e and (ii) v is never activated in e.

For any configuration ρ, let us denote by $P_v(\rho) = (v_0 = v, v_1 = prnt_v, v_2 = prnt_{v_1}, \ldots, v_k = prnt_{v_{k-1}}, p_v = prnt_{v_k})$ the maximal sequence of processes following pointers $prnt$ (maximal means here that either $prnt_{p_v} = \perp$ or p_v is the first process such that there $p_v = v_i$ for some $i \in \{0, \ldots, k\}$).

Let us study the following cases:

Case 1: $prnt_v \in V \setminus S_B$ in ρ.

Since $\rho \in \mathcal{LC}$, $prnt_v$ satisfies $spec(prnt_v)$ in ρ and in any execution starting from ρ (by Lemma 4). Hence, $prnt_v$ is never activated in e. If v does not satisfy $spec(v)$ in ρ, then we have $level_v \neq met(level_{prnt_v}, w_{v,prnt_v})$ or $dist_v \neq 0$ in ρ. Then, v is continuously enabled in e and we have a contradiction between assumption (ii) and the strong fairness of the scheduling. This implies that v satisfies $spec(v)$ in ρ. The fact that $prnt_v$ is never activated in e and that the state of v is consistent with the one of $prnt_v$ ensures us that v is never enabled in any execution starting from ρ. Hence, $spec(v)$ remains true in any execution starting from ρ. This contradicts the assumption (i) on e.

Case 2: $prnt_v \notin V \setminus S_B$ in ρ.

By the assumption (i) on e, we can deduce that there exists infinitely many configurations ρ' such that a process of $P_v(\rho')$ is enabled (since $spec(v)$ is false only when the state of a process of $P_v(\rho')$ is not consistent with the one of its parent that made it enabled). By

construction, the length of $P_v(\rho')$ is finite for any configuration ρ' and there exists only a finite number of processes in the system. Consequently, there exists at least one process which is infinitely often enabled in e. Since the scheduler is strongly fair, we can conclude that there exists at least one process which is infinitely often activated in e.

Let A_e be the set of processes which are infinitely often activated in e. Note that $v \notin A_e$ by assumption (ii) on e. Let $e' = \rho' \dots$ be the suffix of e which contains only activations of processes of A_e. Let p be the first process of $P_v(\rho')$ which belongs to A_e (p exists since at least one process of P_v is enabled when $spec(v)$ is false). By construction, the prefix of $P_v(\rho'')$ from v to p in any configuration ρ'' of e remains the same as the one of $P_v(\rho')$. Let p' be the process such that $prnt_{p'} = p$ in e' (p' exists since $v \neq p$ implies that the prefix of $P_v(\rho')$ from v to p counts at least two processes). As p is infinitely often activated and as any activation of p modifies the value of $level_p$ or of $dist_p$ (at least one of these two variables takes at least two different values in e'), we can deduce that p' is infinitely often enabled in e' (since the value of $level_{p'}$ is constant by construction of e' and p). Since the scheduler is strongly fair, p' is activated in a finite time in e', that contradicts the construction of p.

In the two cases, we obtain a contradiction with the construction of e, that proves the result. □

Let $\mathcal{LC}^*$ be the following set of configurations:

$$\mathcal{LC}^* = \{\rho \in C \,|(\rho \text{ is } S_B^*\text{-legitimate for } spec) \wedge (IM_{m_k}(\rho) = true)\}$$

Note that, as $S_B^* \subseteq S_B$, we can deduce that $\mathcal{LC}^* \subseteq \mathcal{LC}$. Hence, properties of Lemmas 11 and 12 also apply to configurations of $\mathcal{LC}^*$.

Lemma 13 *Any configuration of $\mathcal{LC}^*$ is $(n\Pi(k,\delta)\Delta D, \Pi(k,\delta)\Delta D, S_B^*, n-1)$-TA time contained for spec.*

Proof Let ρ be a configuration of $\mathcal{LC}^*$. As $S_B^* \subseteq S_B$, we know by Lemma 4 that any process v of $V \setminus S_B$ satisfies $spec(v)$ and takes no action in any execution starting from ρ.

Let v be a process of E_B. By Lemmas 11 and 12, we know that v takes at most $\Pi(k,\delta)\Delta D$ actions in any execution starting from ρ. Moreover, we know that v satisfies $spec(v)$ after its last action (otherwise, we obtain a contradiction between the two lemmas). Hence, any process of E_B takes at most $\Pi(k,\delta)\Delta D$ actions and then, there are at most $n\Pi(k,\delta)\Delta D$ S_B^*-TA-disruptions in any execution starting from ρ (since $|E_B| \leq n$).

By definition of a TA time contained configuration, we obtain the result. □

Lemma 14 *Starting from any configuration, any execution of $\mathcal{SSMAX}$ reaches a configuration of $\mathcal{LC}^*$ in a finite time.*

Proof Let ρ be an arbitrary configuration. We know by Lemma 10 that any execution starting from ρ reaches in a finite time a configuration ρ' of $\mathcal{LC}$.

Let v be a process of E_B. By Lemmas 11 and 12, we know that v takes at most $\Pi(k,\delta)\Delta D$ actions in any execution starting from ρ'. Moreover, we know that v satisfies $spec(v)$ after its last action (otherwise, we obtain a contradiction between the two lemmas). This implies that any execution starting from ρ' reaches a configuration ρ'' such that any process v of E_B satisfies $spec(v)$. It is easy to see that $\rho'' \in \mathcal{LC}^*$, that ends the proof. □

Theorem 12 *$\mathcal{SSMAX}$ is a $(n\Pi(k,\delta)\Delta D, S_B^*, n-1)$-TA strongly stabilizing protocol for spec.*

Proof This result is a direct consequence of Lemmas 13 and 14. □

5 Concluding Remarks

We discuss now about the relationship between TA strong and strong stabilization on maximum metric tree construction. We characterize by a necessary and sufficient condition the set of assigned metric that allow strong stabilization. Indeed, properties on the metric itself are not sufficient to conclude on the possibility of strong stabilization: we must know information about the considered system (assignation of the metric).

Informally, it is possible to construct a maximum metric tree in a strongly stabilizing way if and only if the considered metric is strongly maximizable and if the desired containment radius is sufficiently large. More formally,

Theorem 13 *Given an assigned metric $\mathcal{AM} = (M, W, mr, met, \prec, wf)$ over a system S, there exists a $(t, c, n-1)$-strongly stabilizing protocol for maximum metric spanning tree construction with a finite t if and only if:*

$$\begin{cases} (M, W, met, mr, \prec) \text{ is a strongly maximizable metric, and} \\ c \geq max\{0, |M(S)| - 2\} \end{cases}$$

Proof We split this proof into two parts:
1) Proof of the "if" part: Denote $(M, W, met, mr, \prec)$ by $\mathcal{M}$ and assume that $\mathcal{M}$ is a strongly maximizable metric and that $c \geq max\{0, |M(S)| - 2\}$. We distinguish the following cases:

Case 1: $|M(S)| = 1$ (and hence $c \geq 0$).
Denote by m the metric value such that $M(S) = \{m\}$. For any correct process v, we have $\mu(v, r) = \underset{b \in B}{min_{\prec}}\{\mu(v, b)\} = m$. We can deduce that it is equivalent to construct a maximum metric spanning tree for $\mathcal{M}$ and for $\mathcal{NC}$ over this system. By Theorem 4, we know that there exists a $(t, 0, n-1)$-strongly stabilizing protocol for this problem with a finite t, that proves the result.

Case 2: $|M(S)| \geq 2$ (and hence $c \geq |M(S)| - 2$).
By Theorem 12, we know that there exists a $(n\Pi(k, \delta)\Delta D, S_B^*, n-1)$-TA-strongly stabilizing protocol $\mathcal{P}$ for maximum metric spanning tree construction in this case. Denote by Υ the only fixed point of $\mathcal{M}$. Let v be a correct process such that $v \in S_B^*$.

By definition of S_B^*, we have: $\mu(v, r) \prec \mu(v, b)$ for at least one Byzantine process b. As $\mathcal{M}$ is strictly decreasing and has only one fixed point, we can deduce that $\Upsilon \preceq \mu(v, r)$ and then $\mu(v, b) \neq \Upsilon$.

Assume that $d(v, b) > c \geq |M(S)| - 2$. As $\mathcal{M}$ is strictly decreasing, has only one fixed point Υ, and $\mathcal{M}$ has $|M(S)|$ distinct metric values over S, we can conclude that $\mu(v, b) = \Upsilon$. This contradiction allows us to conclude that there exists a process b such that $d(v, b) \leq c$ for any correct process which belongs to S_B^*.

In other words, $S_B^* = \left\{ v \in V | \underset{b \in B}{min}\{d(v, b)\} \leq c \right\}$ and $\mathcal{P}$ is in fact a $(n\Pi(k, \delta)\Delta D, c, n-1)$-strongly stabilizing protocol, that proves the result with $t = n\Pi(k, \delta)\Delta D$.

2) Proof of the "only if" part: This result is a direct consequence of Theorem 8 when we observe that $|M(S)| \leq |M|$ by definition. □

We can now summarize all results about self-stabilizing maximum metric tree construction in presence of Byzantine faults with the above table. Note that results provided in this paper fill all gaps pointed out in related works.

	$\mathcal{M} = (M, W, mr, met, \prec)$ is a maximizable metric
(c, f)-strict stabilization (for any c and f)	Impossible ([15])
(t, c, f)-strong stabilization (for $0 \leq f \leq n-1$ and a finite t)	Possible $\iff \begin{cases} \mathcal{M} \text{ is a strongly maximizable metric, and} \\ c \geq max\{0, \lvert M(S)\rvert - 2\} \end{cases}$ (Theorem 13)
(A_B, f)-TA strict stabilization (for any f and $A_B \subsetneq S_B$)	Impossible ([5])
(S_B, f)-TA strict stabilization (for $0 \leq f \leq n-1$)	Possible ([5] and Theorem 11)
(t, A_B, f)-TA strong stabilization (for any f and $A_B \subsetneq S_B^*$)	Impossible (Theorem 9)
(t, S_B^*, f)-TA strong stabilization (for $0 \leq f \leq n-1$ and a finite t)	Possible (Theorem 12)

To conclude about results presented in this paper, we must bring some precisions about specifications. We chose to work with a specification of the problem that consider the *dist* variable as a O-variable. This choice may appear strong but it seems us necessary to keep the consistency of results. Indeed, impossibility results of Section 3 can be proved with a weaker specification that does not consider the *dist* variable as a O-variable (see [8]). On the other hand, we need the stronger specification to bound the number of disruptions of the proposed protocol. We postulate that our protocol is also TA strongly stabilizing with the weaker specification but we do no succeed to bound exactly the number of disruptions.

The following questions are still open. Is it possible to bound the number of disruptions with the weaker specification? Is it possible to perform TA strong stabilization with a weaker daemon? Is it possible to decrease the number of disruptions without loose the optimality of the containment area?

References

[1] Ariel Daliot and Danny Dolev. Self-stabilization of byzantine protocols. In Ted Herman and Sébastien Tixeuil, editors, *Self-Stabilizing Systems*, volume 3764 of *Lecture Notes in Computer Science*, pages 48–67. Springer, 2005.

[2] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, 1974.

[3] Shlomi. Dolev. *Self-stabilization*. MIT Press, March 2000.

[4] Shlomi Dolev and Jennifer L. Welch. Self-stabilizing clock synchronization in the presence of byzantine faults. *J. ACM*, 51(5):780–799, 2004.

[5] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. The impact of topology on byzantine containment in stabilization. In *Proceedings of DISC 2010*, Lecture Notes in Computer Science, Boston, Massachusetts, USA, September 2010. Springer Berlin / Heidelberg.

[6] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. On byzantine containment properties of the min+1 protocol. In *Proceedings of SSS 2010*, Lecture Notes in Computer Science, New York, NY, USA, September 2010. Springer Berlin / Heidelberg.

[7] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. Bounding the impact of unbounded attacks in stabilization. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2011.

[8] Swan Dubois, Toshimitsu Masuzawa, and Sébastien Tixeuil. Self-Stabilization, Byzantine Containment, and Maximizable Metrics: Necessary Conditions. Research report (available at http://hal.inria.fr/inria-00577062/pdf/duboismasuzawatixeuil.pdf), 03 2011.

[9] Mohamed G. Gouda and Marco Schneider. Stabilization of maximal metric trees. In Anish Arora, editor, *WSS*, pages 10–17. IEEE Computer Society, 1999.

[10] Mohamed G. Gouda and Marco Schneider. Maximizable routing metrics. *IEEE/ACM Trans. Netw.*, 11(4):663–675, 2003.

[11] Shing-Tsaan Huang and Nian-Shing Chen. A self-stabilizing algorithm for constructing breadth-first trees. *Inf. Process. Lett.*, 41(2):109–117, 1992.

[12] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982.

[13] Toshimitsu Masuzawa and Sébastien Tixeuil. Bounding the impact of unbounded attacks in stabilization. In Ajoy Kumar Datta and Maria Gradinariu, editors, *SSS*, volume 4280 of *Lecture Notes in Computer Science*, pages 440–453. Springer, 2006.

[14] Toshimitsu Masuzawa and Sébastien Tixeuil. Stabilizing link-coloration of arbitrary networks with unbounded byzantine faults. *International Journal of Principles and Applications of Information Science and Technology (PAIST)*, 1(1):1–13, December 2007.

[15] Mikhail Nesterenko and Anish Arora. Tolerance to unbounded byzantine faults. In *21st Symposium on Reliable Distributed Systems (SRDS 2002)*, page 22. IEEE Computer Society, 2002.

[16] Sébastien Tixeuil. *Algorithms and Theory of Computation Handbook, Second Edition*, chapter Self-stabilizing Algorithms, pages 26.1–26.45. Chapman & Hall/CRC Applied Algorithms and Data Structures. CRC Press, Taylor & Francis Group, November 2009.