

# Virtual Tree: a Robust Overlay Network for Ensuring Interval Valid Queries in Dynamic Distributed Systems\*

Roberto Baldoni, Silvia Bonomi, Adriano Cerocchi, and Leonardo Querzoni

Sapienza Università di Roma  
Via Ariosto 25, 00185 Roma, Italy,  
{baldoni, bonomi, cerocchi, querzoni}@dis.uniroma1.it,

Today's large scale distributed systems are characterized by strong dynamics caused by the inherent unreliability of their constituting elements (e.g. process and link failures, processes joining or leaving the system). This continuous dynamism has a strong negative impact on distributed algorithms designed to work on such systems. Regular registers [1], Replication [2], in-network aggregation [3], are all examples of such problem.

Considering the in-network aggregate query answering problem in large scale dynamic systems, a precise semantics was introduced by Bawa et al. in their seminal work [4]. One of the semantics introduced in that work, namely *Interval Validity* (IV), requires the answer to *contain at least contributions from all the processes that remain in the system from the moment the query is issued, until the last answer is collected*. This kind of semantics plays a fundamental role in many applications as it prevents contributions coming from correct processes, remaining inside the system, to be eclipsed by transient errors and failures. The same work also proved the impossibility of enforcing interval validity as long as churn (i.e. the rate at which processes can crash/leave or join the system) is unbounded. However, practical experience shows that many systems undergo a continuous (in time) but limited (in its strength) level of churn, and that this level can be reasonably predicted by analytical assessment or by direct measurements [5]. By exploiting this aspect, it would be thus possible to circumvent the previously cited impossibility result and provide interval valid answers to in-network aggregate queries.

On the basis of these motivations, this paper presents the first solution for in-network aggregate query processing that is able to provide query answers complying with the *interval validity* semantics in large-scale dynamic systems with bounded churn. The correctness of our approach is supported by both formal proofs and an extensive set of simulation-based experiments that, due to space constraints, are reported in a separate technical report [6].

***System model and the interval validity semantics.*** A dynamic distributed system is characterized by the continuous arrival and departure of processes (i.e. *churn* phenomenon). We assume the *infinite arrival model* (as defined

---

\* This work has been partially supported by the EU project BLEND and the Italian project DOTS-LCCI.

in [7]) where, in each run, infinitely many processes  $\Pi = \{\dots, pi, pj, pk\dots\}$  may join/leave the system, but at each time unit  $t$ , the distributed system is effectively composed only by a subset of the population, denoted as  $\Pi(t)$ , including all the processes that have joined but have not yet left the system.

Processes can communicate only by exchanging messages on top of perfect point-to-point channels. Each process  $p_i$  has a *partial knowledge* of the system population, i.e. it maintains a *local view* of processes to communicate with. At each time instant  $t$ , the system can be represented as a graph  $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} = \Pi(t)$  and there is an edge  $e_{i,j} \in \mathcal{E}$  connecting two processes  $p_i, p_j \in \mathcal{V}$  if process  $p_j$  is in the local view of process  $p_i$  and vice-versa. We assume a *relaxed asynchronous model*, i.e., there exist known upper bounds on (i) process execution speeds, (ii) message transmission delays and (iii) clock drift rates. In particular, we assume the existence of a known universal maximum delay  $\delta$  on the communication channels. Note that, in this setting, processes can reliably monitor their neighborhoods using a simple heartbeat-based mechanism.

At time  $t_0$ ,  $|\Pi(t_0)| = N_0$ . At time  $t_1$ , processes start joining and leaving the system. We distinguish between (i) *in-churn* ( $\lambda(t)$ ), and (ii) *out-churn* ( $\mu(t)$ ), representing the percentage of processes that join/leave the system at the same time  $t$ . Knowing  $\lambda(t)$  and  $\mu(t)$ , the number of processes that join and leave at each time unit is respectively represented by  $\lambda(t) \cdot N_0$  and  $\mu(t) \cdot N_0$ . We assume that churn is continuous, i.e. it does not exist a time instant  $t$  after which churn ends. A generic configuration of the system  $\mathcal{V}_i$  is defined by the set of processes belonging to the system at a certain point in time. During its lifetime the system is characterized by a totally ordered sequence of configurations. Two successive configurations in this sequence,  $\mathcal{V}_i$  and  $\mathcal{V}_{i+1}$ , differ only for one process that either left or joined the system. The sequence of configurations is an abstraction representing a serialization of the evolution of  $\Pi$  in time due to the effects of churn. Given a query and the sequence  $V = \{\mathcal{V}_i, \mathcal{V}_{i+1}, \dots, \mathcal{V}_{i+j}\}$  of all the configurations experienced during the query execution, the interval validity property [4] can be defined as follows:

**Definition 1.** *A query is said to be interval valid if its result is calculated on a set of processes  $H$  such that  $\cap_{x \in [i, i+j]} \mathcal{V}_x \subseteq H \subseteq \cup_{x \in [i, i+j]} \mathcal{V}_x$ .*

Intuitively, interval validity is satisfied when the result is calculated considering at least all contributions coming from the set of processes that remained in the system during the whole query execution; contributions from processes that leave/join the network while the query is running are not necessarily required.

**The Virtual Tree Architecture.** In order to run an in-network query answering protocol, an overlay network connecting processes in the system, must be defined. Three-shaped topologies offer some clear advantages with respect to other solutions: (i) low diameter (useful to quickly broadcast the query and collect its results), (ii) good scalability and (iii) the possibility to easily define query protocols with clear stopping conditions. However, three-shaped topologies are strongly susceptible to faults and dynamism, a problem that can severely affect

the correct functioning of an in-network query answering protocol. In order to provide query answers complying with the IV property, in fact, two necessary conditions [4] must be met: (1) the overlay network must always be connected and (2) any process that does not leave the system during the query execution must have a stable path (a path that does not change) that connects it to the query source.

We solve the first problem proposing an overlay network topology named the *Virtual Tree (VT)* graph that exploits process clustering to improve its resilience to system dynamics. In order to address the second problem we design an overlay management protocol (OMP) that migrates processes at runtime from the lower layers of the *VT* graph to the upper ones in order to let churn impact its leaves. Through this technique the OMP can guarantee, as long as churn is bounded by a given constant, that the *VT* graph will remain connected and its paths will be stable. On top of these two building blocks we deploy a simple in-network query processing algorithm that provides interval valid answers.

*Virtual Tree graph*: a *VT* graph is constituted by *virtual nodes* [2, 8] (*VN*) and *virtual links* (*VL*) arranged in a tree-shaped topology. A *VN* is constituted by a set of processes interconnected by a full graph (i.e. a *clique*). A *VL* connecting two *VNs* is constituted by the set of links connecting any pairs of processes pertaining to the two different *VNs*. Processes pertaining to two adjacent *VNs* form a completely connected subgraph of the *VT* graph.

*Overlay Management Protocol*: the OMP has two fundamental goals: (i) it must position joining processes in the *VT* graph and (ii) it must guarantee that only *VNs* representing leaves of the tree will possibly disappear as a consequence of a process leave. This last requirement stems from the observation that if a leaf *VN* disappears, none of the *VNs* still present in the graph will see a change in the paths that connect them to the root *VN*, thus the stability of paths connecting *VNs* will be guaranteed. These two goals are reached by arranging processes in the *VNs* such that at any point in time any *VN* (with the exception of leaf *VNs*) is constituted by a number of processes that is possibly above a minimum given threshold  $N_{min}$ . When the size of a *VN* falls below  $N_{min}$  the OMP starts to attract processes from its children *VNs* and moves them in the father process to reconstitute its “safe” size. The  $N_{min}$  threshold is a function of the maximum allowed churn rate and, intuitively, is calculated with the aim of giving “enough time” to the OMP to migrate processes from the lower levels of the graph toward a non-leaf *VN* that is currently experiencing a local churn surge. New processes joining the system can be accommodated in any *VN* (a join, in fact, cannot negatively impact IV) with size smaller than a threshold  $N_{max}$  or, alternatively in a new leaf *VN*. The  $N_{max}$  threshold is needed to limit the amount of generated overhead that grows exponentially with the size of a *VN*.

*In-network query processing algorithm*: the query processing algorithm we propose is a simple adaptation of a broadcast/convergecast approach with partial result aggregation, modified to run on the *VT* graph topology. Starting from the leaf *VNs* partial results are aggregated in intermediate *VNs* and forwarded

to the upper levels until they reach the root of the *VT* graph. The absence of disconnections in the *VT* graph and the stability of virtual paths (both provided by the OMP), together with the structure of the query protocol guarantee that the returned result will include contributions from all physical processes that remained in the system for the whole query duration and will thus comply with the IV property. Note that, *having a VT graph*, the capability of providing IV results only depends from the connectivity of the system. This intuition is formalized by the following theorem:

**Theorem 1.** *Let  $p_i$  be the process issuing a query  $q$  at time  $t$  and let  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  be a *VT* graph at time  $t$ . If  $\mathcal{G}$  is always connected then  $q$  terminates and satisfies the interval validity semantics.*

*Virtual Tree Overlay Network Connectivity.* Connectivity of a In *VT* graph is deterministically guaranteed as long as the out-churn is lower than a certain threshold. The following Lemma reports this bound assuming that  $T_{move}$  represents the time needed to let a process move from a child *VN* to its father *VN*.

**Lemma 1.** *Let  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  be a *VT* graph at time  $t_0$ . If, for any time  $t$ ,*

$$\sum_{i \in [t, t+T_{move}+1]} \mu(i) < N_{min}/N_0$$

*then the *VT* graph is connected.*

## References

1. R. Baldoni, S. Bonomi, A.-M. Kermarrec, and M. Raynal, "Implementing a register in a dynamic distributed system," in *ICDCS*. IEEE Computer Society, 2009, pp. 639–647.
2. S. Dolev, S. Gilbert, N. A. Lynch, E. Schiller, A. A. Shvartsman, and J. L. Welch, "Virtual mobile nodes for mobile ad hoc networks," in *Proceedings of the 18th International Conference on Distributed Computing*, 2004, pp. 230–244.
3. P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in *Proceedings of the Second International Conference on Mobile Data Management*, 2001.
4. M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani, "The price of validity in dynamic networks," *Journal of Computer and System Sciences*, vol. 73, no. 3, pp. 245–264, 2007.
5. S. Saroiu, K. P. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking (MMCN)*, January 2002.
6. R. Baldoni, S. Bonomi, A. Cerocchi, and L. Querzoni, "Virtual tree: a robust overlay network for ensuring interval valid queries in dynamic distributed systems," MIDLAB 4/2011 - Università degli Studi di Roma "La Sapienza" - <http://www.dis.uniroma1.it/~midlab/publications.php>, Tech. Rep., 2011.
7. M. Merritt and G. Taubenfeld, "Computing with infinitely many processes," in *Proceedings of the 14th International Conference on Distributed Computing*, 2000.
8. I. Eyal, I. Keidar, and R. Rom, "Distributed clustering for robust aggregation in large networks," in *Proceedings of the 5th Workshop on Hot Topics in System Dependability*, 2009.