A Modular Approach to Optimizing Highly–Dynamic Distributed Systems

Extended Abstract

Benoît Garbinato[⋆] Fernando Pedone[†] Rodrigo Schmidt[†]

[⋆]Université de Lausanne, CH-1015 Lausanne, Switzerland
Phone: +41 21 692 3409 Fax: +41 21 692 3405
E-mail: benoit.garbinato@unil.ch

[†]École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland
Phone: +41 21 693 4797 Fax: +41 21 693 6600
E-mail: {fernando.pedone, rodrigo.schmidt}@epfl.ch

1. Dynamic Distributed Systems

With the emergence of a mobile and large-scale Internet, highly-dynamic distributed systems are becoming increasingly important. Examples of this growing importance can be found in recent researches in large-scale peer-to-peer protocols [1, 3], as well as in ad hoc network technologies [4].

Intuitively, a *highly-dynamic* distributed system can be defined as a system whose configuration changes *very often*. What may vary in a configuration depends on the considered model. In a probabilistic crash-recovery model, for instance, the configuration might consist of a probability associated with each node, representing the average uptime of that node.

2. Devising Optimal Algorithms

Devising optimal algorithms in such systems is challenging because optimality can usually only be defined in terms of a configuration *given a priori*, via some *cost function* (also known as *objective function*) to minimize. Formally, an algorithm solving some distributed problem is said to be *optimal with respect to cost function* f if it also solves some optimization problem of the form given by Equation 1. In this classical formulation of optimization problems, f is the cost function to minimize, x is a vector of system parameters and $\mathcal{S}$ is the *feasible set* expressed as one or more constraints on the given configuration. Note that cost function f is usually expressed in terms of the same a priori configuration.

$$\begin{array}{ll} minimize & f(\vec{x}) \\ subject\ to & \vec{x} \in \mathcal{S} \end{array} \tag{1}$$

To illustrate the challenge of devising optimal algorithms in highly-dynamic distributed systems, let us take our previous example of a probabilistic crash-recovery model. Assume that we want to devise a probabilistic reliable broadcast algorithm that is optimal with respect to the total number of generated messages. That is, we want to reach every node with at least probability K, while generating a *minimum number of messages*. Using the formalism of Equation 1, we define vector $\vec{x}$ as the number of (re-)transmissions through each link, cost function f as the total number of messages, and the feasible set $\mathcal{S}$ as the values of $\vec{x}$ for which the probability to reach all nodes is greater than or equal to K. If in addition the configuration (i.e., node crash probabilities in our example) changes frequently, our probabilistic broadcast algorithm has to address two key issues: (1) detecting configuration changes, and (2) adapting its behavior to those changes.

3. A Modular Approach

To address these issues, we propose a modular approach that *insulates* the design of optimal algorithms from the burden of adapting to configuration changes.

More precisely, our approach consists in addressing each issue in a distinct layer. In a first layer, we devise an algorithm that solves the distributed problem we want to solve and that is optimal *given exact knowledge about the system configuration*. At this level, we only have to prove that our algorithm is optimal assuming such exact knowledge. In a second layer, we devise an algorithm that builds up knowledge of the system configuration and gives it to the first layer. At this level, we have to show that our algorithm eventually converges toward the actual system configuration. So, if we can prove that the algorithms of both layers satisfy their respective requirements, their assembling will result in an *adaptive optimal distributed algorithm*. The idea is that whenever a configuration change occurs, the second layer will eventually converge to it, and as soon as this happens, the first layer will exhibit optimal behavior.

4. Some Examples

In [2], we apply this approach to probabilistic reliable broadcast. Our optimal algorithm relies on the assumption that each node knows the topology and the reliability of nodes and links, and uses this knowledge to minimize the number of messages needed to reach all nodes with a given probability. This is achieved by having each node first compute a *Maximum Reliability Tree* (MRT) of the system. The MRT is a spanning tree containing the most reliable paths connecting all nodes. We calculate the MRT using a modified version of Prim's algorithm [2].

In our adaptive protocol, in addition to determining the MRT, nodes constantly try to approximate the topology and the reliability of nodes and links. If these remain stable "long enough", our adaptive protocol converges toward the optimal one. Initially, nodes know only the links connecting them directly to their neighbors. To share this knowledge, each node periodically sends heartbeat messages with its view of the topology to all its neighbors. When receiving a heartbeat, a node updates its local knowledge with the information received and eventually learns the global system topology. Heartbeats are also used to determine the reliability of nodes and links.

In a different context, we consider applying this modular approach to routing in peer-to-peer overlay networks. The well-known proximity neighbor selection (PNS) strategy selects the closest neighbors of a node in the underlying topology to build its routing table [3]. However, this approach does not necessarily lead to the best global route between two peers. Moreover, when it is used in tree topologies, the restriction imposed on the paths makes the delay between nodes increase exponentially along routes [1], aggravating the harm of an early bad decision. The cost involved in keeping track of the closest peers in large-scale highly-dynamic systems also prohibits the usage of perfect PNS in real systems and, therefore, some simpler heuristics must be used to approximate it [3].

Our approach can be used to address this problem in two ways. First, it can be used to build the topology efficiently. Second, during routing a node can forward a message to the neighbor that will lead to the shortest global path to the destination. Consider initially that nodes have a full view of the system membership, the links between nodes, and the estimated transmission delays of these connections. In this case, a node that wants to join the overlay can choose the best way to connect itself to the topology in order to keep its optimality (e.g., if it is a tree, keeping it balanced). Moreover, while routing a message, a node can find the best route locally using some shortest-path algorithm and forwarding the message through the best global path.

Following our modular approach, an adaptive protocol has to maintain the topology information in each node and make it converge toward the actual system configuration. The strategy could be similar to the one used in [2], with periodic heartbeat messages exchanged between nodes to propagate topology information. Putting optimal and adaptive algorithms together, we get a complete adaptive protocol for building the topology and routing messages in dynamic peer-to-peer overlays. Finally, to improve the scalability of the solution, we are considering grouping nodes together and propagating different levels of topology information.

References

[1] M. Castro, P. Druschel, Y. Charlie Hu, and A. Rowstron. Proximity neighbor selection in tree-based structured peer-to-peer overlays. Technical Report MSR-TR-2003-52, Microsoft Research at Cambridge, 2003.

[2] B. Garbinato, F. Pedone, and R. Schmidt. An adaptive algorithm for efficient message diffusion in unreliable environments. In *Proceedings of IEEE International Conference on Dependable Systems and Networks conference (DSN'2004)*, June 2004.

[3] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proceedings of the ACM SIGCOMM 2003*, Karlruhe, Germany, Aug. 2003.

[4] C. Perkins. *Ad Hoc Networking*. Addison-Wesley, 2000.