Abstract. As a consequence of the rise of cloud computing, the reliability of network protocols is gaining increasing attention. However, formal methods have revealed inconsistencies in some of these protocols, e.g., Chord, where all published versions of the protocol have been discovered to be incorrect. Pastry is a protocol similar to Chord. Using TLA+, a formal specification language, we show that LuPastry, a formal model of Pastry with some improvements, provides correct delivery service. This is the first formal proof of Pastry where concurrent joins and lookups are simultaneously allowed. In particular, this article relaxes the assumption from previous publication to allow arbitrary concurrent joins of nodes, which reveals new insights into Pastry through a final formal model in TLA+, LuPastry. Besides, this article also illustrates the methodology for the discovery and proof of its invariant. The proof in TLA+ is mechanically verified using the interactive theorem prover TLAPS.

Keywords: Formal verification · Interactive theorem proving · Network protocols

1 Introduction

1.1 The Pastry Protocol

Pastry ([16], [3], [4]) is a structured P2P algorithm realizing a Distributed Hash Table (DHT, by [5]) over an underlying virtual ring. The network nodes are assigned logical identifiers from an ID space of naturals in the interval $[0, 2^M - 1]$ for some $M$. The ID space is considered as a ring\(^1\) as shown in Fig. 1, i.e. $2^M - 1$ is the neighbor of 0.

The IDs are also used as object keys, such that an overlay node is responsible for keys that are numerically close to its ID, i.e. it provides the primary storage for the hash table entries associated with these keys. Key responsibility is divided equally according to the distance between two adjacent nodes. If a node is responsible for a key we say it covers the key.

The most important sub-protocols of Pastry are join and lookup. The join protocol eventually adds a new node with an unused network ID to the ring.

\(^1\) The ring here does not refer to algebraic group structure with operation.
The lookup protocol delivers the hash table entry for a given key. This paper focuses on the correctness property \textit{CorrectDelivery} (mentioned as dependability in algorithm paper [3]), requiring that there is always at most one node responsible for a given key. This property is non-trivial to obtain in the presence of concurrent join or departure of nodes, i.e., \textit{churn}. To cope with that, each Pastry node maintains a local state of a set of nodes called \textit{leaf sets}, as shown in Fig. 1, consisting of a left set and a right set of the same length, which is a parameter of the algorithm. The nodes in leaf sets are updated when new nodes join or failed nodes are detected using a maintenance protocol. A Pastry node also maintains a routing table to store more distant nodes, in order to achieve efficient routing.

In the example of Fig. 1, node \textit{a} received a lookup message for key \textit{k}. The key is outside node \textit{a}’s coverage. Moreover, it doesn’t lie between the leftmost node and the rightmost node of its leaf sets. Querying its routing table, node \textit{a} finds node \textit{b}, whose identifier matches the longest prefix with the destination key and then forwards the message to that node. Node \textit{b} repeats the process and finally, the lookup message is answered by node \textit{c}, which covers the key \textit{k}. In this case, we say that node \textit{c} \textit{delivers} the lookup request for key \textit{k}.

\subsection*{1.2 The Methodology}

\textit{TLA$^+$} by [6], is a formal specification language based on untyped \textit{Zermelo-Fraenkel} (ZF) set theory with choice for specifying data structures, and on the Temporal Logic of Actions (TLA) for describing system behavior. It is chosen to analyze and verify the correct delivering and routing functionality of Pastry, because it provides a uniform logic framework for specification, model-checking and theorem proving. It fits protocol verification quite nicely, because its concept of actions matches the rule/message-based definition of protocols. In addition,