July 13, 2017

Construction of a Temporal Tree and its Coarse Grain View in a Streaming Big Data Context

Crina Cristiana Arsenie
Hewlett Packard Enterprise

Christophe Girardin
Hewlett Packard Enterprise

Construction of a Temporal Tree and its Coarse Grain View in a Streaming Big Data Context

Abstract: Structuring data as a tree can apply in every day's life (like friends, family network of connections etc.), but also in industries like Telecommunications. With the fast pace of technology advancements, these structures are growing, changing, bringing therefore a dynamicity that has to be managed. This disclosure provides an efficient technique to construct trees in a Streaming Big Data context and to optimize the processing of these tree updates in real time by other consumer applications.

The technique presented in this disclosure is domain-agnostic, therefore it can be plugged on any external system, whose models of components are structured as trees and can change constantly at runtime, and send this components through a bus in streaming to any consumer application.

This disclosure proposes the following techniques as a solution for the two problems identified:

1. Construction of trees in streaming while doing a validation of the tree completion according to a model. Therefore, a check of the tree consistency during time is implemented. The trees are stored in a Big Data DB.
2. Consumer applications have to retrieve the trees during time, with minimum cost. Therefore, this disclosure provides a coarse grain view of the tree's lifecycle to optimize its processing by consumer applications.

In the two Figures a case where we apply the technique presented in this disclosure in a Telecom context is described, on tree structures modeling services. The technique presented in this disclosure is domain agnostic and consists of:

Input (Model) in streaming (Figure 1)

The model is structured as a tree built dynamically in streaming. The updates, creates and deletes of nodes or edges are defined by transaction. Per transaction, a bulk of messages is received in streaming. Each message is about: a node (vertex), an edge, nodes or edges properties. Each message represents an atomic information: create (i.e. create of a node), delete (i.e. delete of an edge), update (i.e. update of a node).

Controller Component (Figure 2)

This component consists in consuming the input model (tree structure received in streaming) to construct the tree in a coherent way, while managing its lifecycle and keeping the history in a Big Data DB. It also provides a coarse grain view. To solve the problems posed, the Controller component consists of the two following main modules (see the Figure 2):

The input ingestion module (saves the data in the Staging area):

This module consists of loading the messages which come in streaming through the Bus, in tables in the DB, so that they can be processed after by the SQL Batches module which is responsible to construct the tree and its coarse grain view. In order to realize this, the tables in the DB need to be filled in by this module for the creation, deletion and updates of the nodes and edges, as seen in

the Figure 2. A transaction contains a bulk of messages corresponding to a set of created, deleted, updated nodes and edges. A transaction will be taken into account if it is complete, i.e. all its messages are loaded into the tables. Another table is created to manage this information called SYNCHRONIZATION table.

In our architecture, the messages are being received through streaming and are loaded into the Big Data DB so that they can be processed after in a fast and efficient way by the SQL Batches. For the implementation part, a streaming bus can be used like Kafka and for the Big Data DB a column Big Data DB.

SQL Batches module (constructs the final tree and coarse grain view in the Processing area)

This module is responsible of constructing the tree. This means that firstly, the consistency of the bulk of messages has to be checked (Messages synchronization per transaction). Then, putting together the different parts of the tree received by bulk (Building of the tree at time t) has to be done. It is also responsible of building the coarse grain view of the tree (Coarse grain output).

The tree is constructed each time the batch runs (each x time). For this, by taking the tables filled in by the ingestion module and also by the tree already constructed at time t-x, we can build the tree at time t and save it in the following tables:

DIM_NODE: this table contains the nodes of the tree in their latest state (time t), by merging the information about the create, delete and update per node

DIM_TREE: this table contains the tree edges constructed in their latest state (time t), containing the edges between elements of {DIM_NODE}

These tables are filled in by the result of the SQL statements being executed in the batch.

This module is also responsible of producing the coarse grain output, which represents a synthetic information of the tree's lifecycle. This means that it highlights the critical nodes in the tree, i.e. the nodes who have a father or a son who suffered a topology modification (delete, create, update) either between two transactions, either between two batch executions. For this, SQL statements are being executed in the batch and fill in the result in the table DIM_COARSE_GRAIN using the following algorithm:

Algorithm to discretize by time (each batch run at time t):

The node n is critical in creation case:

If n is created and is the root of a tree at time t

If n is created and is a son of a node existing at time t-x

The node n is critical in update case:

If n exists at time t-x and has a son that is created or deleted at time t

The node n is critical in delete case:

If n is deleted and was the root of a tree at time t-x

If n is deleted and was a son of a node z at t-x and node z is not deleted at time t

This algorithm can also be discretized by transaction, by adding at each step a check on the transaction numbers.

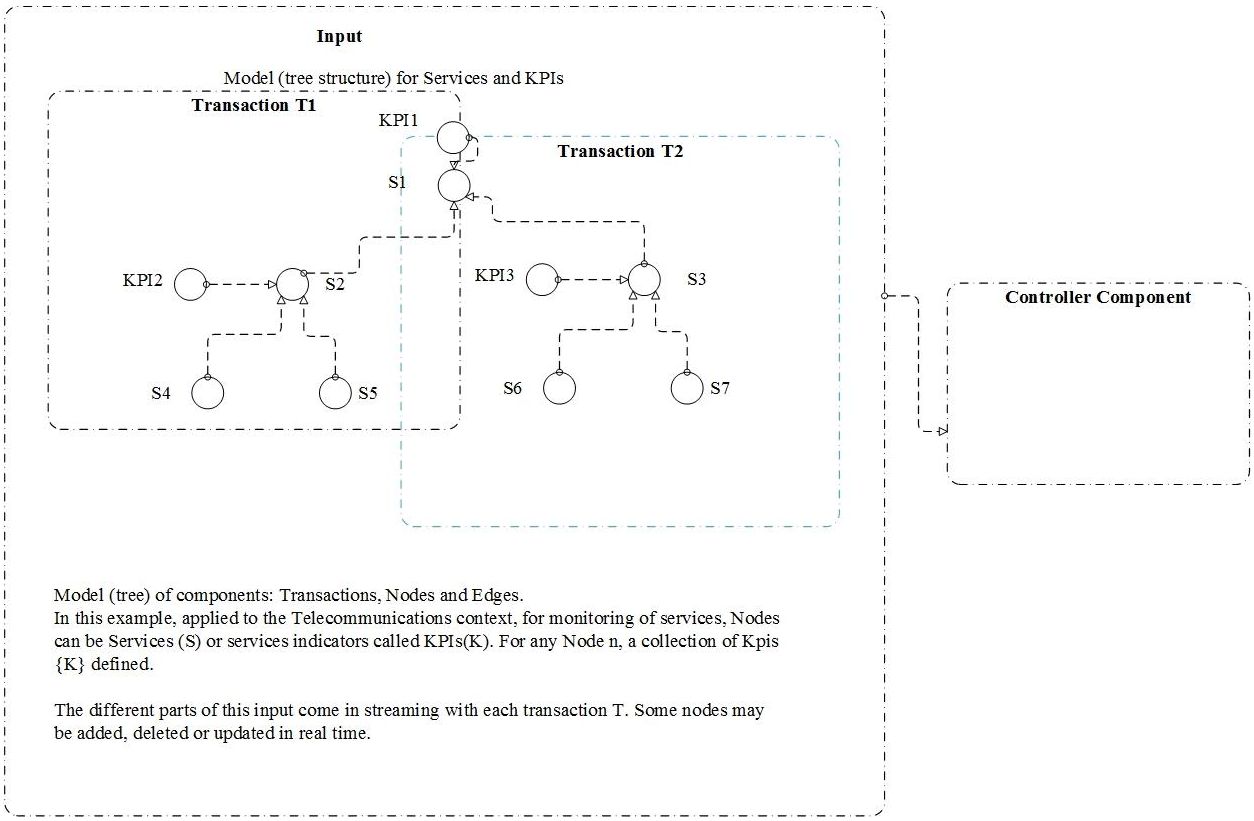

Figure 1 Input

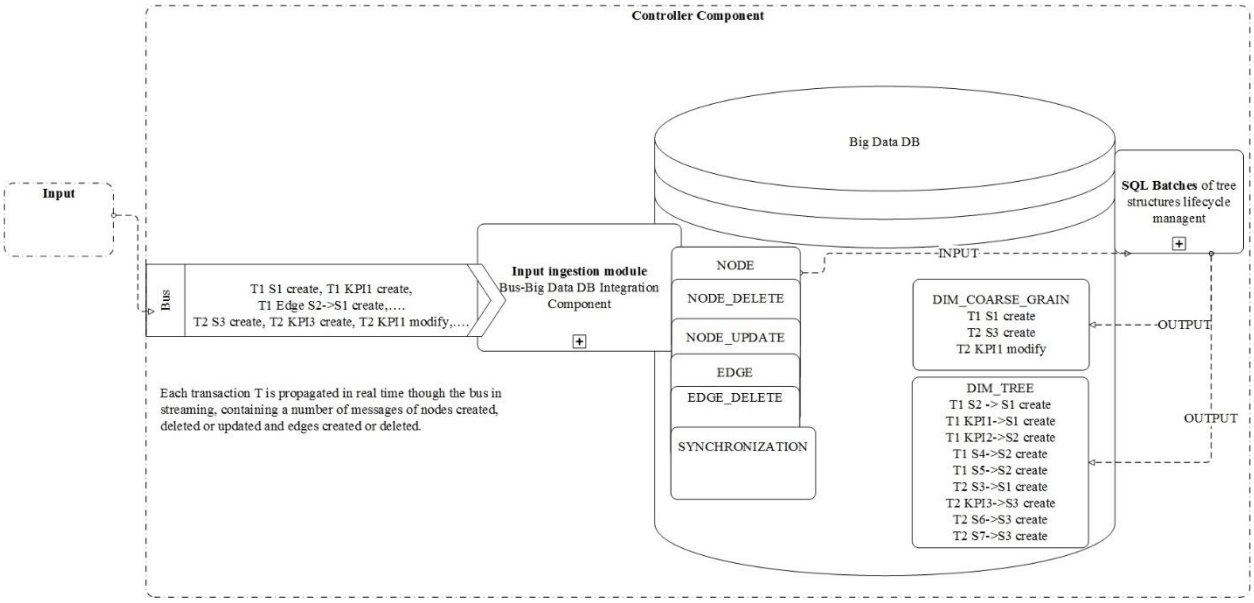

Figure 2 Controller Component

Disclosed by Crina Cristiana Arsenie and Christophe Girardin, Hewlett Packard Enterprise