Frames

Traditional window definitions are static and simplistic. They allow users to segment data streams in terms of time and tuple count. While adequate for many applications, time and tuple-based window definitions lack the sophistication needed to support more complex applications. We present Frames, a dynamic , adaptable and expressive method for segmenting data streams. Frames allow dynamic, adaptable stream segmentation so that framing schemes can adapt to changing data distributions as is inevitable in a long-running data stream query. Frames are expressive so that they can better capture user query needs and produce more accuracy with, in many cases, significantly fewer results.

*Example 1* – Traffic Speed Averages: Traffic archives receive traffic speed and count data at regular intervals from sensors in roadways. In Portland, OR such data is received at 20-second intervals. A common query over this stream is to request the “current” speed at a particular location. Using windows, the “current” speed is typically calculated using a 3- or 5-minute rolling (sliding) window. The purpose of the window is to smooth out variability in the 20-second data. Reporting the “current” speed directly from the 20-second readings would result in a jittery “current” speed, so rolling averages are used to smooth out the jitter. However, we observe that different lengths of averages are appropriate at different times of day. When traffic volume is high, we may wish to use a shorter rolling average, whereas when traffic volumes are lower, such as in overnight time periods, we may wish to use a longer rolling average. While windows cannot adapt to changing traffic volumes, frames can.

*Example 2 – “Spiky” Data:* Data sets collected from sensors or other sources vary in their data distributions. Some data sets have a “spiky” pattern – wherein data is relatively stable for a long time with intermittent sharp peaks as shown in Figure XXXX. In many cases, as in this figure, the width of the peaks is “small” relative to the lengths of time where the data is relatively stable. A commonly-desired option on such data sets is to detect the start and end of the peaks; in some applications, this detection must be done in real time. Consider the options for using windows to detect such peaks. As the widths of the peaks are “small” relative to their interval of occurrence (in common language, peaks are small and sparse), then small windows – with a window length similar to the width of the peaks are ideal for detecting the peaks. However, using small windows will result in a large number of window results, most containing only the information that the data is still in a stable state. If window size is increased, the number of results will reduce; however, peaks may be lost (undetected) due to averaging effects or the start/end time of the peaks may be obscured due to the long window lengths. Frames, in contrast, can provide relatively accurate start/end times of the peaks without the clutter of unnecessary results when the data is stable. Could list examples of data sets that tend to have “spiky” data.

*Example 3 – Detection of River Water Mixing:* Oceanographic scientists study the fluid movements of water in the Columbia River. One method of study is to release dye into the river at a particular location and to measure the spread of the dye through the water. The dye spread is measured by driving a boat around in the river and towing a device that measures dye intensity through the water; the device moves up and down through various depths of the water and in this way the spread of the dye is recorded. A Figure XXX shows a plot of XX vs. YY (some interesting dye plot). The end product of the measurements is a histogram showing density versus location of dye as shown in Figure XXX. Current procedure calls for calculating average dye intensity, temperature and salinity for each meter of depth of water. Accomplishing this calculation with windows seems near impossible. Frames can accomplish this easily and perhaps do better by framing on density and/or dye intensity.

-----------------------------------------------------------------------------------------------------------------------------

Frames

Traditional window definitions are static and simplistic. They allow users to segment data streams in terms of time and tuple count. While adequate for many applications, time and tuple-based window definitions lack the sophistication needed to support more complex applications. We present Frames, a dynamic , adaptable and expressive method for segmenting data streams. Frames allow dynamic, adaptable stream segmentation so that framing schemes can adapt to changing data distributions as is inevitable in a long-running data stream query. Frames are expressive so that they can better capture user query needs and produce more accuracy with, in many cases, significantly fewer results.

Example 1: Traffic. Traffic archives receive traffic speed and count data at regular intervals from sensors in roadways. In Portland, OR such data is received at 20-second intervals. A common query over this stream is to request the “current” speed at a particular location. Using windows, the “current” speed is typically calculated using a 3- or 5-minute rolling (sliding) window. The purpose of the window is to smooth out variability in the 20-second data. Reporting the “current” speed directly from the 20-second readings would result in a jittery “current” speed, so rolling averages are used to smooth out the jitter. However, we observe that different length averages are appropriate at different times of day.

Example 2: Spiky data