Big Data Space Fungus

M. Kersten
CWI Amsterdam, The Netherlands

1. MOTIVATION

The big data questions addressed in the database community remind me of the famous fable of the invention of Chess. "When the creator of the game of chess (in some tellings an ancient Indian mathematician, ...) showed his invention to the ruler of the country, the ruler was so pleased that he gave the inventor the right to name his prize for the invention. The man, who was very wise, asked the king this: that for the first square of the chess board, he would receive one grain of wheat (in some tellings, rice), two for the second one, four on the third one, ..."[wikipedia]

The Big Data hype resembles the data scientist king who fanatically fills the squares of the board. Furthermore, it appears as is if the computer scientists and engineers acts as the faithful civil servants, to go after their treasure chest, putting on the board ever more powerful, increasing piles of hardware. Every 1.5 year we double the amount of data and processing power. A futile activity as the fable has clearly identified thousands of years ago.

The corollary of the chess board payment scheme using rice (or wheat) is that not only you can not find enough of it in the universe, you will also never be able to consume the amount collected. The lesson is *Don't collect more rice (wheat) you can eat, otherwise it will rot away in storage*. This also shows the way out of the data deluge dilemma. We should recognize and respect the natural laws of data *freshness* and *rotting* in the core of our Big Data analytics offerings.

2. BIG DATA SPACE ROTTING

For the sake of brevity consider a single table $R(t, f, A_1, ..., A_n)$ where A_i denotes the attributes, t the real world time it was inserted, and a freshness property $f \in (0.0 - 1.0)$ initially set to 1.0. So far, the relation R can be used like a normal relational table. Next we let nature come into the game, our first natural law for Big Data .

The extend of table R decays with a periodic clock of T seconds using a data fungus F until it has been completely disappeared

Each tuple in R decays over time, reflected in an ever decreasing freshness value. When the freshness reaches zero, the tuple is discarded from R. An old-fashioned decay function F would be to consider retention times, where after the data will be discarded. However, many more data fungi can be considered, based on their rate of decay, what to decay, how to decay.

To illustrate consider a simple fungus, say 'EGI' (Evict Grouped Individuals), which runs as follows. At each clock cycle T:

- select an element from R inversely randomly correlated with its age and seed it with the fungi F, decreasing its freshness.
- select all F infected elements and decrease their freshness, also affecting the direct neighboring tuples at equal rate.

EGI creates rotting spots in R, which leads to removing complete insertion ranges when not being taking care of by its owner. The speed by which it decays could comes both from the initial infection at a certain time stamp t_j, but also the bi-directional growth along the time axes, infecting neighboring tuples. The effect of EGI is similar to Blue Cheese, where portions of the cheese turns into its rotting equivalent over time. It remains edible for a long time though.

3. BIG DATA SPACE FRESHNESS

The evident approach to avoid rotten data is to cook it into useful information a.s.a.p. It is a task normally undertaking by the data ingestion pipeline. Here too we can take nature one step further by focusing on the queries over R. Consider the select-from-where queries $A = Q(T, R, P)$, the query Q over table R with predicate P and target expression T. The answer set is A. The second natural law for Big Data becomes:

The extent of table R is replaced by each query Q into the union of the answer set of Q and the reduced extent of R

This rule stresses the point that once you take something out of R, you should distill it into useful knowledge, summary, consumed by the user, or stored in a new container subject to different data fungi. All tuples in R satisfying P are discarded immediately.

The database is kept in optimal health condition if you regularly can turn rotting portions into summaries for later consumption, or inspect them once before removal.

4. CONCLUSIONS

This short note carves out a barren landscape of innovative research and technology. It addresses the seemingly uncontrollable data explosion using two natural laws for Big Data. They should become an integral part in the toolkit of the data scientist.

Beware that the steps proposed are nowadays part of data science pipelines, and even fundamental to streaming database systems, or Complex Event Processing systems

Of course, our approach does not rule out technology progress on the road already taken. Aside from the trend towards better harnessing hardware to create bigger 'fridges', we should find better (datamining) 'cooking' schemes to discard/avoid the rotten data.