Lesson 4

Finding Hidden Patterns

Collapsing Infinity to a Point

Wholeness of the Lesson

Human brain is a pattern matching machine. Therefore our problem solving techniques are based on finding hidden patterns buried deeper in Nature. In this context, boils down to finding hidden patterns buried deeper in Big Data.

A successful action results from a deeper dive into silence, into pure intelligence, just as, in archery, the arrow flies truer and hits its mark more consistently if it is pulled back farther on the bow.

PAIRS AND STRIPES

One common approach for synchronization in MapReduce is to construct complex keys and values in such a way that data necessary for a computation are naturally brought together by the execution framework.

This section introduces two common design patterns known as “pairs" and “stripes“.

Example - Building co-occurrence matrix

Event. Any thing we are interested in can be thought of as the event. We can represent an event using integer.

Example: A customer bought the following books.

Gone With the Wind // 145 be its ID

Titanic // 329 be its ID

Life of Pi // 278 be its ID

…

This can be represented as

145 329 278 …

Example - Building co-occurrence matrix

Point 1:

Thus sequence of events associated with one customer can be represented in one record.

Point 2: Two different records belong to two different customers.

Example:

5 7 8 2 9 // belongs to one customer

8 9 7 8 2 // belongs to another customer

Example - Building co-occurrence matrix

Concept of a window

Given a event, you can define a window based on the problem at hand.

Examples

8 9 7 8 7 2

Example 1. Window is just two events following the event different from the current event.

W(8) = {9, 7}, W(9) = {7, 8}, W(7) = {8, 2}, W(8) = {7, 2}, W(7) = {2}, W(2) = {}

Example - Building co-occurrence matrix

Examples (cont'd)

8 9 7 8 7 2

Example 2.

W(x) = {y | y appear after x and before another x or the end of the list}

W(8) = {9, 7}, W(9) = {7, 8, 7, 2}, W(7) = {8}, W(8) = {7, 2}, W(7) = {2}, W(2) = {}

More examples of window definition

Window(x) = {y | y appear after x in the same sentence}

Window(x) = {y | y is the next two items after x}

Window(x) = {y | y appear after x in the same chapter}

Window(x) = {y | y happened within two weeks after x}

Example - Building co-occurrence matrix

Formally, the co-occurrence matrix is a square n × n matrix where n is the number of events.

A cell m_{ij} contains the number of times j appear in all windows of i.

Example - Building co-occurrence matrix

Let W(x) = {y | y appear after x and before another x or the end of the list}

Example

8 9 7 8 7 2

7 2 8 4 2 1

m_{72} is 3. That is, 2 appear in all windows of 7, 3 times.

Main Point 1

The concepts, Event and Window together gives a layer of abstraction that makes the algorithms useful in a wide variety of situations. Every abstraction captures the central theme and thus makes it more useful. *All manifested objects can be abstracted to the Self.*

Relative frequencies (Pair)

Partition Rule.

The partition must just depend upon the u of the key (u, v).

```
int getPartition(key k, int numReducers)
    return Math.abs(k.left.hashCode())% numReducers
```

Relative frequencies (Pair)

```
int compareTo(Object O1, Object O2)
    k = compareTo(O1.left, O2.left)
    if (k != 0) return k
    return compareTo(O1.right, O2.right)
```

Pairs approach for co-occurrence matrix

```
class Mapper
    method Map(docid a, doc d)
        for all term u in record r do
            for all term v in Window(u) do
                Emit((u, v), 1) .
class Reducer
    method Reduce(Pair(u, v), Integer [c1, c2, …])
        s = 0
        for all Integer c in [c1, c2, …] do
            s = s + c .
        Emit((u, v), s)
```

Stripe approach for co-occurrence matrix

```
class Mapper
   method Map(docid a,  doc d)
      for all term u in record r do
         H = new AssociativeArray
         for all term v in Window(u) do
              H{v} = H{v} + 1 .          //Tally words co-occurring with u
         Emit(u, H)
class Reducer
   method Reduce(term u, AssociativeArray [H1, H2, …])
      H_FINAL  = new AssociativeArray
      for all stripe H in [H1, H2, …] do
          H_FINAL = H_FINAL + H                    //elementwise addition
      Emit(u, H_FINAL)
```

Main Point 2

In order to find hidden patterns from the historical data without assuming anything or depending on theories such as probability, we can use the co-occurrence matrix. The basis of all knowledge is our ability to find and formalize patterns. *"Water the root, enjoy the fruit" is an example of the knowledge abstracted from the hidden patterns.*

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

1. A co-occurrence matrix is a powerful tool to unearth hidden patterns.

2. In order to be useful in a wide variety of situations, the concept of neighbor is introduced.

---

3. ***Transcendental consciousness:*** *is the source of all patterns in life.*
4. ***Impulses within the Transcendental Field:*** *It is these impulses are an abstraction of all activities.*
5. ***Wholeness moving within itself::*** *In unity consciousness the individual functions spontaneously in an effortless manner.*