Parallel Solution for Near Repeat Analysis

Dr. Haihang You, Kenneth McKanders

OVERVIEW

There is a growing consensus that many socioeconomic dynamics are spatially concentrated, such as unemployment and violence. Recently, researchers have recognized that spatial and temporal factors in crime events should be addressed jointly instead of being treated separately. Near Repeat calculation is designed to reveal the correlation of these factors between events; however, during the examination of an event chain with multiple events, the computational complexity of Near Repeat calculation increases exponentially. This research is focused on the development of a parallel solution for Near Repeat computation, as well as a fast algorithm for multi-event Near Repeat calculation. In particular, this study will be conducted using randomly generated datasets of events that consist of x and y coordinates and timestamps.

INTRODUCTION

Given a list of events represented as a tuple containing the X and Y coordinates, as well as the time of the event, the range of time, and the operational range (distance), the algorithm should return N_{11}, N_{12}, N_{21}, N_{22} where:

N_{11} is defined as the set of points within the specified space-time:

$$N_{11} = |(i,j) d(i,j) \le d \text{ and } t(i,j) \le t|$$

N_{12} is defined as the set of points within the specified space, but outside of the specified time:

$$N_{12} = |(i,j) d(i,j) \le d \text{ and } t(i,j) > t|$$

N_{21} is defined as the set of points outside of the specified space, but within the specified time:

$$N_{21} = |(i,j) d(i,j) > d \text{ and } t(i,j) \le t|$$

N_{22} is defined as the set of points outside of the specified space-time:

$$N_{22} = |(i,j) d(i,j) > d \text{ and } t(i,j) > t|$$

In order to generate these numbers, events must be compared to each other to determine their relationship. When events are related, they create a cluster within the dimensions of the space-time area. Events within this cluster have a certain probability that they were performed by the same person. Since this probability deprecates over time (needs reference), it is therefore understood that a wider time specification means that it is less likely that the events are related by the person that performed them.

	$t(i,j) \le t$	$t(i,j) > t$
$d(i,j) \le d$	N_{11}	N_{12}
$d(i,j) > d$	N_{21}	N_{22}

ALGORITM ANALYSIS

Traditionally, when comparing elements in a pairwise manner, a 2-D grid is constructed which shows the relationships for each possible pair. While this is fine for smaller datasets, the space requirement for larger datasets far exceeds not only the allowable space for a program (usually 2GB), but also most standard RAM sizes. Therefore, a new storage method is required for these relations.

Additionally, the algorithmic complexity for finding these pairwise relations is $O(n(n-1)/2)$, and increases exponentially for the number of elements in the relation set; this means that a new relation function is required to reduce the calculation complexity for sets of higher numbers.

METHODS

First, the size issue was handled. The original storage method was an $n \times n$ grid. There are two things known about storing relations in this way:

- The elements along the diagonal of the grid represent the relations between any event and itself
- The grid is mirrored about this same diagonal

With this knowledge, we can simplify the grid into a single array in the following manner:

:	1	2	3	4
1	Related	Related	Distance	Time
2	1. Related	Related	Distance	Time
3	2. Distance	4. Distance	Related	Unrelated
4	3. Time	5. Time	6. Unrelated	Related

1. Related	2. Distance	3. Time	4. Distance	5. Time	6. Unrelated

The elements are related between the two structures in the following manner:

- $A1xy = B1z \text{ where } z = (\sum j=0 \uparrow min(x,y) \blacksquare n-j-1) + max(x,y) - min(x,y)$

However, this is still not enough reduction for a sparse matrix. The structure can be further reduced into a Compressed Row Storage format.

METHODS cont.

The Compressed Row Storage object (Q) is constructed in the following manner:

- $Q1a = relation1xy$ *x, y are events in the system which are not unrelated*
- $Q1b = yv = \sum j=0 \uparrow min(x,y) \blacksquare n-j-1$, $(x,y) = (v,z-v)$
- $Q1c = val1p$ {$\blacksquare p$=0, val=0p>0, $val1p - val1p-1$ = *amount of elements in column p*

Once this structure has been created in memory, it becomes easier to find relations of elements using the relation:

- $A1xy =$ {$\blacksquare x=y$, *related*$\blacksquare x \neq y$, $Q1a$ [k] *such that* $Q1c$ [min(x, y)]$\le k < Q1c$ [min(x,y) +1], $Q1b$ [k]=max(x,y) −min(x,y)

Given this storage of the pairwise relations between each element, finding relations for any size set becomes a recurrent relation:

- $f(i1, i2, \cdots, ir) =$ {$\blacksquare r$=2, *table lookup2*$< r \le n$, $f(i1, i2, \cdots, ir-1) \wedge \uparrow \blacksquare \wedge j=1 \uparrow r-1 \blacksquare f(ij, ir)$

PRELIMINARY RESULTS

Preliminary testing was done on 5, 15,000, and 150,000 event datasets. Each event was composed of an x and y coordinate, and a timestamp. The distance was measured as the Euclidean distance between two points, and the time as the difference in time between the two given events. Results were generated in serial code.

CONCLUSIONS

The 5-event dataset was written purposely to find relations in time, distance, and a combination of the two. The other datasets, however, were written to find relations between completely random events. While the 5-event set gave expected results, the 15,000- and 150,000-event sets quickly showed that a parallel solution is required for large datasets. Future work will be to finish implementing the program on Kraken, and to run analysis of the 15,000- and 150,000-event sets with relation sets of 3 or more events.

CONTACT INFO

Haihang You
hyou@utk.edu

Kenneth McKanders
kamckanders@gmail.com

REFERENCES

Knox, G. (1964). The detection of space-time interactions. *Applied Statistics* 13:25-29.