Divide and Conquer Algorithm for Determining Sequences Patterns in Spatiotemporal Clustering

Arief Fatchul Huda[1]
Ito Wasito[2]
T. Basaruddin[2]

***Abstract*—Pattern in spatiotemporal data with symbolic data are sequences with the same subsequence from profiles(data). There are four step to determining pattern. Time complexity in step one is linear when use Generalized Suffix Tree (GST) to find out candidate core subsequence. Time complexity in second step are n^2 to merge candidate subsequence and n^2 to find unique elemen, so time complexity is $O(n^2)$. Using divide and conquer algorithm for finding unique elemen, it can reduce time complexity to $O(n \log n)$ where n is number of subsequence.**

I. INTRODUCTION

Spatiotemporal data clustering deal with discovering pattern of similar characteristics from given data. The statistical analysis for spatiotemporal data clustering is well for numerical data. Unfortunately, statistical analysis cannot model symbolic values and poor to handling of string. Another problem with statistical spatial analysis is that computation of the results is expensive[Koperski, 1999]. In this paper, symbolic representation is used to spatiotemporal data. With symbolic representation, we can analyse symbolic and numeric data.

Symbolic data are arranged in sequences. A temporal profile is a sequence of successively observed values in one location (space). Snapshot data representation is use to represent the data. In figure 1, three difference temporal profile measured from three location in four time observed. One time observing represents in one layer.

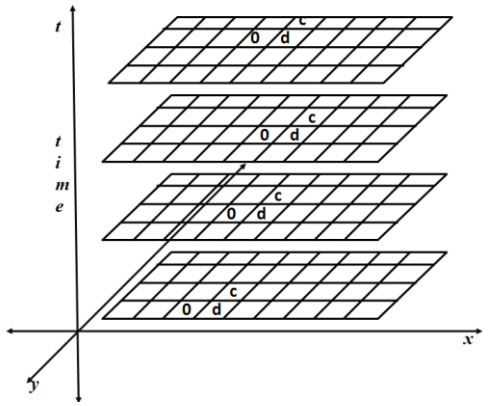

Figure 1. Snapshot data representation, each layer represented time observed and contain space elemen.

Each layer have some grids. A grid is a place of area. Similar patterns of profile are captured and the profile id are clustered which express these patterns on the basis of time and space constrain. A temporal pattern is a subsequence shared by number of profiles. And spatial clustering is the similar temporal profile are close to each other. The profile discovered might not be similar during the whole time period and just occur in a similar manner at some time point. Thus, patterns can represented as subsequences and interesting patterns are those that are shared by maximum number of profile. In other word, the patterns are sets of clusters obtained using a spatial profile applied to the patterns discovered during the temporal clustering Step[Kakkar 2004].

	t1	t2	t3	t4	…
P1	a	c	a	d	…
P2	a	c	d	b	…
P3	b	c	d	b	…
P4	d	b	a	d	…
…					
…					

Figure 2. Temporal profile

II. METHODOLOGI SPATIOTEMPORAL CLUSTERING

We define temporal profile and spatial profile to cluster sequence.

Temporal profile

1. *Mathematic Dept. UIN Sunan Gunung Djati Bandung*
2. *Faculty of Computer Science, Universitas Indonesia*
3. *Informatic Enginering, Universitas Pancasila*

In [Abraham, 1998], Each element of the table is reffered as $D(p_i,t_i)$ where p_i is temporal profile (row) id and t_j is the j observed time point, fig. 2. Each row of the table *D* is a sequence which represents the profile temporal behavior. Temporal pattern is the subsequence of the sequence of temporal profile and a set of profile that contain these subsequences. Some definition of temporal pattern and its measurement are :

- Subsequence *S* containing random choice of columns (m time points) t_1, t_2, …,t_m in *D*.
- A set *R* containing *k* distinct rows (r_1, r_2, …, r_k)
- A formula to determine distance between two profiles ri and rj where I,j = 1,2,…,k is

$$dist(ri,ri)= \Sigma_{t\in s}|D(r_i,t).D(r_j.t)| \quad ..(1)$$

where $D(r_i,t).D(r_j.t) = 1$, if $D(r_i,t)=D(r_j.t)$, else $D(r_i,t).D(r_j.t)=0$.

Spatial profile

To define spatial pattern from spatial profile we add position coordinates x, and y of temporal profile in the cluster. Let $R_g = \{r_1, r_2, \dots, r_k\}$ be the set of profiles (rows) that are temporal pattern contain k distinct rows. Formally, for each temporal pattern, we define spatial pattern,

- Each set S contains distinct row (profile) id's where *i*= 1, 2, 3, …, *k* and *k* = |*Rg*|
- A metric to determine distance between two profile r_1 and r_2 is defined as :

$$Dist(r_1, r_2) = \sqrt{((x_2-x_1)^2+(y_2-y_1)^2)} \quad \dots(2)$$

where (x_1,y_1) are coordinates of $r_1$and (x_2,y_2) are coordinates of r_2.

Each set S_i consists of all those elements from R_g such that dist(r_i, r_k) between two rows (temporal profile) r_1 and r_2 is below or equal to some threshold, i.e dist(r_i,r_k) ≤ T where $k \neq i$.

Quantification and Clustering

There are four step to find out pattern in set of sequences. The detail of each step as below :

Step :

1. Find out the set S of all shared subsequence such that :
   a. Each subsequence *s* in the set *S* occurs in a set of sequences corresponding to the profile (row) id's (p_1, p_2, …, p_k) and the subsequence in each complete sequence start at the same time point
   b. Subsequence *s* and the set of profiles (set of row id's) whose sequences share this subsequence (beginning at the same location) are paired up to give < P,s>
2. Cluster the set of profile P found in Step-1 such that cluster contains a number of overlapping set. Temporal subsequence is formed by the union of subsequences corresponding to all the set of profiles icluded in a cluster. This temporal subsequence characterizes the cluster and is called the core of the cluster[46].
3. Generalize the core subsequences developed in Step 2 to search the temporal profile for more interesting pattern.
4. Discover S_i, the set of profiles wich are close to each other.

Determining core of cluster is to determine subset of all subsequence (Step 1) which

$$|P_1 \cap P_2| \;/\; |P_1 \cup P_2| >= T \quad \dots(3)$$

where P_i is subsequence in Step 1 and T is user defined threshold.

Time complexity to merge subsequence is n^2. After merging proses, there are a lot of duplicate subsequence. Time complexity to find unique elemen is n^2, too. In this paper we propose Divide and Conquer algorithm to reduce time complexity of finding core pattern become O(*n* log *n*).

s	size(s)	Set R	|R|
aba-eb-	5	{0,2}	2
abc---d	4	{1,4}	2
ab---bd	4	{1,2}	2
ab----d	3	{1,2,4}	3
ab-----	2	{0,1,2,4}	4

Figure 3. Core subsequence

Generalization is written in reguler expression or regex. A reguler expression is a way to define pattern of characters. Each symbol in regex is considered replaceble by its immediately preceding or following alphabet. For example, in the core subsequence aba-***eb-***, we accept alphabet '***a***', '***b***' in place of '***a***' and '***a***', '***b***', '***c***' in place of '***b***', and '***d***', '***e***','***f***' in place of '***e***'. Then we find profile id's that has subsequence of regex of core subsequence. This problem need 3^n time for regex of each subsequence, where n is number of character in subsequence.

III. Divide and Conquer Algorithm

Divide and conquer (D&C) is an algorithm strategy based on multi-branched recursion. A divide and conquer algorithm breaking down a problem recursively into two or more sub-problems of the same (or related) type, until these become simple enough to be conquer(solve) directly. Then combine the solution of the sub-problems to give a solution on the original problem.

Divide and Conquer in Step 2

In Step 1, we get the candidate core subsequence. Step 2, we merge all candidate core subsequence to get core subsequence. Step to find out core subsequence from candidate subsequence are :

a) Merge subsequence P_i and P_j, where i,j = 1, 2, .., n which eq. 1.
b) To find unique subsequence from step 1, we use divide and conquer strategy.

Three part of Divide and Conquer to find core subsequence in step 2b, are :

- **Divide** : divide the n-duplicate core subsequence into n/2 duplicate core subsequence.
- **Conquer** : one subsequence is solution
- **Combine** : add each subsequence into table and use only one subsequence for all the same core subsequence

Algorithm step 2a

```
Function Step2a(TbSubP1,TbP1)→ TbSubP2a
m = size(TbP1)
T = 0,5
for i = 1 to m
  for j = i+1 to m
    add=(TbP1(i)∩TbP1(j))/(TbP1(i)∪TbP1(j))≥T
    if add
      TbSubP2a = merge(TbsubP1(i),TbsebP1(j))
return TbsubP2a
```

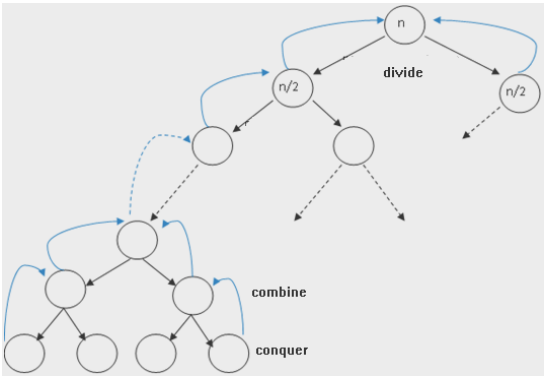

Figure 4. Divide and Conquer of determining core subsequence

Algorithm step 2b

```
function unique(TbsubP2a)→TbsubP2b
sTb2a = size(TbSubP2a);
InTb = TbSubP2a;
  if sTb2a == 1 or sTb2a == 0
    TbOut = Tb;
    return
  else

    m    = size(Tb);
    Tb1 = unique(InTb(1:floor(m/2),:));
    Tb2 = unique(InTb(floor(m/2)+1:m,:));

   m1=size(Tb1);
   m2=size(Tb2);
```

```
catat =zeros(m1,m2);
Tb3 = Tb2;
baris = m2;
for ii = 1 to m1
  catat1 = 0;
  for jj = 1 to m2
     if Tb1(ii,:)== Tb2(jj,:)
        catat1=1;
  if catat1 == 0
     baris = baris + 1;
     Tb3(baris, : ) = Tb1(ii,:);
TbSubP2a = Tb3;
Return TbSubP2a
```

IV. EXPERIMENTAL

In this experiment we use simulation data, i.e five sequences and each of sequence consist 7 character/symbol. Alphabet of sequence are *a*, *b*, *c*, *d*, *e*, *f*, and 0.

bcbdfcd bcde0ce
bcbbfce f0b0ace
bcdfabe dcd0bea

and coordinate (*x*,*y*) are

(1,1), (1,2),
(2,1), (2,2),
(3,1), (3,2)

Snapshot view of this spatiotemporal data are in figure

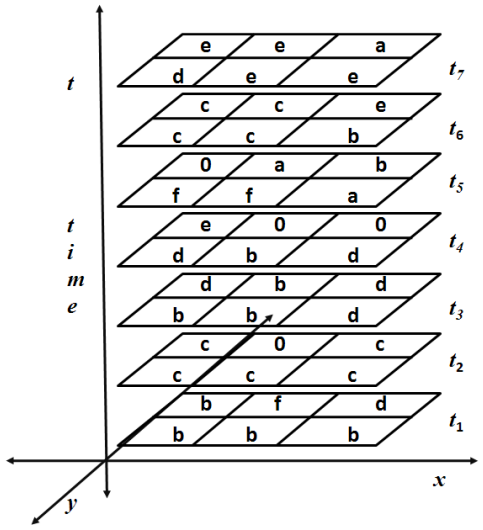

Figure 5. Snapshot view of the spatiotemporal data.

Step 1 resulting 113 subsequences. We are using subsequences that are consisted in 2 or more profile id's. There are 15 subsequences, see figure 5.
Merging subsequence in phase 2a, we have 56 subsequences with duplicates subsequences. In figure 6, row 2, 5, 12, and 15 contains same subsequences.
We use devide and conquer to remove duplicate subsequence and we get 34 unique subsequence. Time complexity to find unique element is $O(n \log n)$ if we use Divide and Conquer algorithm. It is more effisient than we use naïve algorithm that has time complexity $O(n^2)$, i.e. we must compare one by one subsequence.
Pattern of temporal profile are discovered in this step. We can see the id of profile that have same pattern. In

figure .. , profile 1 dan 3 have same pattern in t_1-t_3 and t_5-t_6.

Subseq	Size	Profile id	|R|
b------	1	{1,2,3,5}	4
b c-----	2	{1,2,3,5}	4
b c b----	3	{1,3}	2
b c d----	3	{2,5}	2
-c-----	1	{1,2,3,5}	4
-c b----	2	{1,3}	2
-c d----	2	{2,5}	2
--b----	1	{1,3,4}	3
--d----	1	{2,5}	2
----f--	1	{1,3}	2
----a--	1	{4,5}	2
----f c-	2	{1,3}	2
-----c-	1	{1,2,3,4}	4
-----c e	2	{2,3,4}	3
------e	1	{2,3,4,5}	4

Figure 6. Candidate subsequence from step 1

No	Subseq	Profile id
1	b c-----	{1,2,3,5}
2	b c b----	{1,3}
3	b c d----	{2,5}
4	b c-----	{1,2,3,5}
5	b c b----	{1,3}
6	b c d----	{2,5}
7	b-d----	{2,5}
8	b---f--	{1,3}
9	b---f c-	{1,3}
10	b----c-	{1,2,3}
11	b-----e	{2,3,5}
12	b c b----	{1,3}
13	b c d----	{2,5}
14	b c-----	{1,2,3, 5}
15	b c b----	{1,3}
16	b c d----	{2,5}
..	…	….

Figure 7. Merging candidate subsequence with duplicate subsequence

No	Pattern	size	id	|R|
1	b c b-f c-	5	{1,3}	2
2	-c b-f c-	4	{1,3}	2
3	b c d---e	4	{2,5}	2
4	b c b--c-	4	{1,3}	2
5	b c b-f--	4	{1,3}	2
6	b c--f c-	4	{1,3}	2
7	--b--c e	3	{3,4}	2
8	--b-f c-	3	{1,3}	2
…	…	…	…	…

Figure 8. Core subsequence from step 2b.

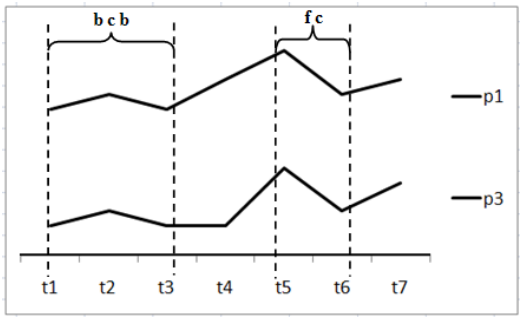

Figure 9. Pattern of profile 1 and 3 have same pattern on t_1-t_3 and t_5-t_6

Spatial clustering discover the profile which have same temporal pattern use euclidean distance. Each profiles has location information, i.e coordinate *x* and *y*. A metric to determine spatial similarity are euclidean distance (eq. 2) and the threshold is user defined.

V. Conclusion and Future Work

There are four step to cluster spatiotemporal data. In each step there some algorithm. In this paper we use divide and conquer to compute step 2, and that can reduce time complexity from $O(n^2)$ to $O(n \log n)$.
The next research are to reduce time complexity in step 3, i.e generalization proses. Reguler expresion is combination of subsequence. Time complexity to make this combination is $O(n^3)$. Before this, we must remove core subsequence that are subsequence from another core subsequence.

References

[1] Abdurrahman A, Pilouk M, Spatial Data Modelling for 3D GIS, Springer-Verlag, Berlin 2008

[2] Abraham T, Roddick J. F., Survey of Spatio-Temporal Databases, GeoInformatica 3:1, 69-99, Kluwer Academic Pub, Boston, 1999

[3] Christakos G, Modern Spatiotemporal Geostatistics, Oxford University Press, 2000

[4] Daniel B. Niel, Detection of Spatial and Spatio-Temporal Cluster, Ph.D Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, 2006

[5] Dorohanceanu B, Nevill-Manning C, A Practical Suffix-Tree Implementatio for String Searching, Dr. Dobb's Journal, 2000.
[6] E. Martin, Toward Spatiotemporal Patterns, Springer, 2004
[7] Esko Ukkonen, On-line construction of suffix trees, Depart of Computer Science, Univ. of Helsinki, 1991
[8] Kakkar, Shagun, Methodology for Clustering Spatio-Temporal Database, Thesis, Univ. Of Cincinnati, 2004
[9] Lawson Andrew B, Denison David G.T., Spatial Cluster Modelling, Chapman & Hall/CRC, 2002
[10] Lawson, Andrew, Spatial Cluster Modeling, Chapman & Hall, London, 2002
[11] Pelekis N, etc, Literature review of spatiotemporal database models, The Knowledge Enginering Review, Vol. 19:3, 235-274, Cambridge University Press, 2004
[12] Pelekis N, Theodoulidis B, Kopanakis I, Theodoridis Y, Literature review of Spatio-Temporal database models, The Knowledge Enginering Review, Vol. 19:3, 235-274, Cambridge Univ. Press, 2004