International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-3 Number-1 Issue-8 March-2013

PLST: Point Location Using Search Technique Algorithms

Syed Jaffar Abbas[1] , Raju Manjhi[2]
Department of Computer Application, Ranchi University, Ranchi, India[1]
Department of Computer Application, Ranchi University, Ranchi, India[2]

Abstract

Search technique is a basic tool of data mining. A hierarchical search technique for point location based on k-d tree is introduced. Algorithms of building and traversing this binary data structure are explained. In this technique we will go over how to use a k-d tree for finding the k nearest neighbours of a specific point or location and then will also go over how to find all neighbours within some distance specified by the user.

Keywords

Data mining, k-d tree, point location, A* algorithm, PLST algorithm

1. Introduction

Point location problem arise on a quite different scale as well. Assume that we want to implement an interactive geographic information system that display a map on a screen. By clicking with the mouse on a country or cities, the user can retrieve information about that country. While the mouse is moved the system should display the name of the country or cities underneath the mouse pointer somewhere on the screen. Every time the mouse is moved, the system has to recomputed which name to display.

K-D Tree definition

- Used for point location and multiple database queries, k –number of the attributes to perform the search.
- To perform search in 2Dimensionalspace .
- Search components (x,y) interchange!

A* Search definition

The A* algorithm integrates in serving to find out into a search procedure. Instead of choosing the next node with the least cost (as measured from the start node), the choice of node is based on the cost from the start node plus an estimate of a heuristic estimation.In the searching, the cost of a node v could be calculated as:

f(v) = distance from s to v + estimate of the distance to d.

= d(v) + h(v,d)

= d(v) + sqrt((x(v)) – x(d))2 + (y(v) – y(d))2)

Where x(v), y(d) and x(v), y(d) are the, coordinates for node v and the destination node d.

2. Related Work

Application of point location

To store and retrieve the information like Name, Address and contact no of hospital in Ranchi. From the specific position, we can find out which one is the nearest hospital. To do this work we have used k-d tree, A* search algorithm and PLST algorithm.

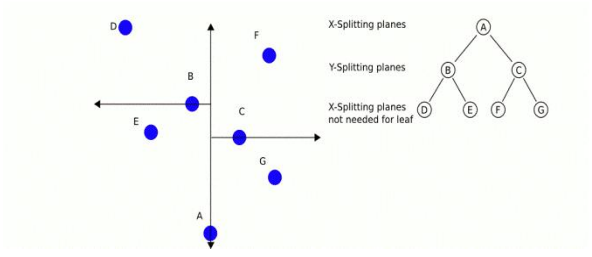


Fig 1: Point location as a node and Binary Tree

Table 1: Information about Hospital

NODE	Name of Hospital	Address	Contact No.
A	APPOLLO	IRBA,Ormanjhi	0651 - 2276041 / 42 / 2275699 / 899
B	RIIMS	Bariatu	0651- 2541533
C	Kashyap Eye Hospital	Purulia Road Ranchi	0651 – 2531255
D	Raj Hospital & Research Centre	Main Road Ranchi	0651- 2331128 , 2330129
E	Guru Nanak Hospital	Station Road ,Ranchi	0651-246 0506
F	Vivekanand Hospital	Krishna Nagar Colony, Ratu Road	0651 - 2282929.
G	Doranda Hospital	Doranda	0651- 2223546

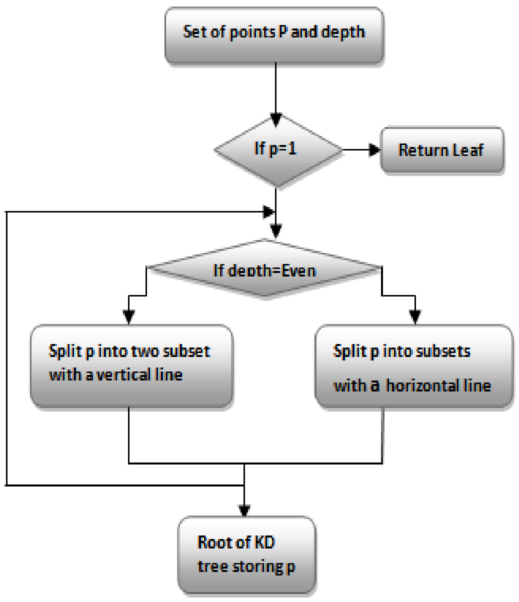
3. Methodology

Suppose any person from outside Ranchi come to Ranchi for treatment. He will try to find out the nearest hospital if its current position i.e Ranchi railway station so from above example nearest hospital is Node no (E) to identify the hospital information. We have used k-d tree to store the information of various hospital in Ranchi at different location. One can easily find out the location of any hospital in Ranchi.

Overview of KD tree and A* search and proposed PLST algorithm

3.1 KD tree algorithm – The process is as follows. At the root we split the set P with a vertical line l into two subset of roughly equal size. The division line is stored at the root. P1, the subset of points to the left or on the splitting line, is stored in the left sub tree and P2, the subset to the right of it, is stored in the right sub tree. In general, we split with a vertical line at nodes whose depth is even, and we split with a horizontal line at nodes whose depth is odd.

Flowchart of KD Tree


Set of points P and depth
If p=1
Return Leaf
If depth=Even
Split p into two subset with a vertical line
Split p into subsets with a horizontal line
Root of KD tree storing p

Algorithm to construct k-d tree

```
Algorithm CONSTRUCTKDTREE(P, d)
Input.A collection of points P and the current depth d.
Output.kd-tree is storing P as a root.
1. ifP contains only one point.
2. then return a leaf which store this point
3. else if d is even
4. thenDivide P into two subsets with a vertical line l to find the median of x-coordinate of the points in P. Let P1 be the set of point to the left of l or on l
, and let P2 be the set of points to the right of l
5. elsedivide P into two subsets with a horizontal line l to find the median of y-coordinate of the points in P. Let P1 be the
set of points below l or on l, and let P2 be the set of points above l.
6. $ν_{leftchild}$← CONSTRUCTKDTREE(P1,d+1)
7. $ν_{rightchild}$← CONSTRUCTKDTREE (P2,d+1)
8. Create a node ν storing l, make $ν_{leftchild}$ the left child of ν, and make
$ν_{rightchild}$ the right child of ν.
9. returnν
```

Algorithm to Search through k-d tree

```
Algorithm TRAVERSINGKDTREE(ν,R)
Input.The root of akd-tree, ν and a range R.
Output.All points at leaves below ν that lie in the range.
1. ifν is a leaf
2. thenReport the point stored at ν if it lies in R.
3. else if area(leftchild(ν)) is fully contained in R
4. thenREPORTSUBTREE(leftchild(ν))
5. else if area(lc(ν)) intersects R
6. thenTRAVERSINGKDTREE(leftchild(ν),R)
7. ifarea(rightchild(ν)) is fully contained in R
8. thenREPORTSUBTREE(rightchild(ν))
9. else if area(rightchild(ν)) intersects R
10. thenTRAVERSINGKDTREE(rightchild(ν),R)
```

3.2 A* Search algorithm - we want to still be able to generate the path with minimum distance

A* is an algorithm that it uses heuristic to guide search and while ensuring that it will compute a path with minimum distance

A* computes the function $f(n) = g(n) + h(n)$

- $g(n)$ = “distance from the starting node to reach n”
- $h(n)$ = “estimate of the distance of the cheapest path from n to the goal node”

Example:

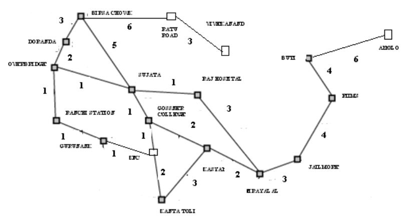


Fig 2: Example of A* graph

A* Search algorithm:

```
Algorithm: A*search(v)
for each u G:
d[u] = infinity;
parent[u] = NIL;
End for
d[s] = 0;
f(V) = 0;
H = {s};
whileNotEmpty(H) and targetNotFound:
u = Extract_Min(H);
label u as examined;
for each v adjacent to u:
if d[v] > d[u] + w[u, v] , then
d[v] = d[u] + w[u, v];
p[v] = u;
f(v) = d[v] + h(v, D);
DecreaseKey[v, H];
```

3.3 The Proposed PLST Algorithm

The PLST algorithm begins by applying the KD tree algorithm upon the set of points p to return vertex or node v, where d≤dv or d>dv, d is the distance from origin and dv is the optimal calculated distance. The observation from fig 1.a and 1.b leads to the following query algorithm i.e. search KD tree algorithm: We traverse the KD tree, but visit only that node v whose region is intersected by the query rectangle. When a region is fully contained in the query rectangle , we can report all the points stored in its sub tree. If d ≤dv then return τ otherwise apply A* search algorithm which is used to find the nearest neighbor as we have shown it in Fig.3. The steps of our proposed PLST algorithm are given below:

PLST Algorithm

Algorithm: PLST(τ, d,d1)

```
1.Apply CONSTRUCTKDTREE(p) algorithms where P is a point
2. Apply CONSTRUCTKDTREE (v,R) where v is the vertices and R is the range.
3. Initialize an empty status structure τ
4. V<- root(τ)
5. While v is not leaf and ($d \leq d_v$ or $d > d_v$ )
6.do if ($d \leq d_v$ )
7.then
returnτ
 8. else call A*search(v) algorithm
 9. return v.
```

4. Result

The PLST algorithm was validated upon three syntheticcases. Our results were compared with that of *A* search* algorithm (best validity values for each cases) and arepresented in Table 3. The PLST algorithm is able toidentify actual distance for all the cases andvalidity values obtained are minimum
D(n)= g(n) +w(n,d) (recursively till the final position f(n)) from Fig.2 and Table-1

Table 2: Result

Case(km)	**Initial Position g(n)**	**Final Position f(n)**	**DistanceD(n)**
1	E	A	24
2	E	A	23
3	E	A	27

5. Conclusion and Future work

In this paper K-D tree have been used in conjunction with A* search algorithm for identifying nearest neighbors using point location . Further we have used PLST algorithm to identify point location . Our future work comprises of validation of the proposed algorithm on real data for scalability and robustness.

Acknowledgment

We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout this research paper. We acknowledge the effort of those who have contributed significantly to our paper. We are highly obliged and thankful to our parents for their motivation. We are on the green grass in expressing our sincere thanks and thankful from bottom of our heart for the gratitude of them, for showing their confidence in us by letting us work on the research paper of this magnitude.

References

[1] Jiawei Han and M. Kamber, Data mining concepts andtechniques, Morgan Kaufmann Publishers, 2007.
[2] M H Dunhum, Data Mining-IntroductoryandAdvanced Topics, Pearson Education, 2007.
[3] Mark de Berg. Otfried Cheong, Marc van Krefeld,Markovermars Computational GeometryAlgorithm and Application .
[4] Spatial Data Structures JianpingFan Department of Computer ScienceUNC-charlotte.

[5] Burroughs, Principles of Geographical Information Systems for Land Resources Assessment, Oxford University Press, 1986.
[6] Samet, The Design and Analysis of Spatial Data Structures and Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS. Addison-Wesley, 1989.
[7] Symposium on Large Spatial Databases (SSD – computer science perspective, geographic information systems, image databases;fullyrefereed papers.
1989 Santa Barbara; 1991 Zurich; 1993 Singapore; 1995 Maine Proceedings in Lecture Notes in Computer Science, Springer-Verlag..
[8] A Static Optimality Transformation with Applications to Planar Point Location John IACONO,WolfgangMulzer, International Journal of Computational Geometry & Applications Vol. 22, No. 04, pp. 327-340.
[9] www.jharkhandtourism.in.

Syed Jaffar Abbas, I was born in Ranchi, Jharkhand India, in 1981. I received the Master degree in Computer Application from the University of JAMIA, Newdelhi, India, in 2007, and the M.Tech. in Computer Science from the Birla Institute of Technology (BIT) Mesra, Ranchi, India respectively. I worked as a Software Engineer in Wipro Technology Ltd in 2008.

Mr. Raju Manjhi, I completed my M.C.A and I also have acquired the degree of M.Tech from BIT ,Mesra ,Ranchi. I have international certification of Microsoft .Net framework 3.5, A.S.P .Net Application development (070-562) with 100%. I am working as a lecturer of P.P.K.College ,Bundu ,Ranchi University, Ranchi. I have been the part of different programs and seminars like the seminar from faculty development programs which was held at ICFAI national college. I have also attended the seminar on “Application and management on green Technology” which was organized ICFAI University Jharkhand.