Technion
Israel Institute of Technology

Efficient Orienteering-Route Search over Uncertain Spatial Datasets

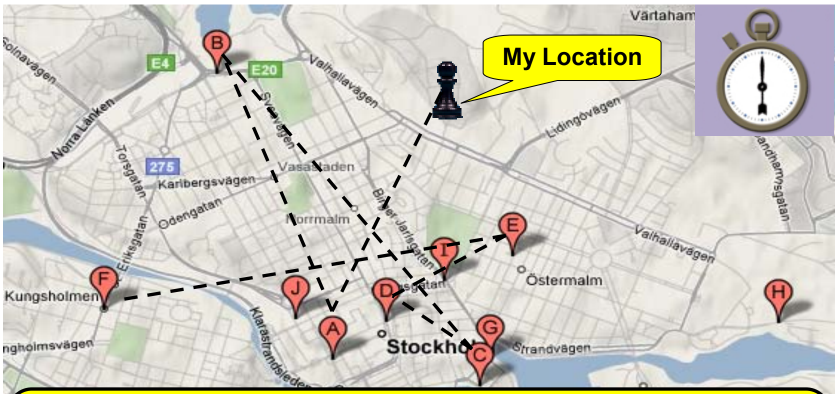
Mr. Nir DOLEV, Israel
Dr. Yaron KANZA, Israel
Prof. Yerach DOYTSHER, Israel

Route Search

- A standard search engine on the WWW returns a list of Web pages ordered according to their relevance to the search terms
- In a geographic search, the user may need to actually visit the discovered entities

Example 1 – A Standard Spatial Search

("Restaurants Stockholm")



Visiting the entities according to their order in the result may not be effective, since a path that goes through the entities may travel back and forth

Route Search

- We suggest an alternative solution, where objects in the search result are ordered in a way that forms a route based on both their relevance to the search and their locations

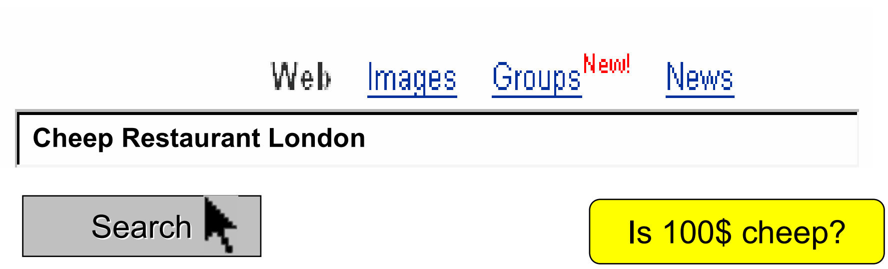
- We consider such a search as a ***route search***

Route Searching Over Uncertain Datasets

- Spatial data is inherently uncertain due to various reasons such as its acquisition process, imprecise modeling and manipulation (e.g., **integration**, incorrect updating and **inexact querying**)
- We represent spatial data uncertainty by attaching to each object a **confidence** value indicating its probability to be correct
- A user may be able test the correctness of an object by visiting the entity at the location of that object

5 FIG Working Week 2008, Stockholm, Sweden

Example 2 – Inexact Queries



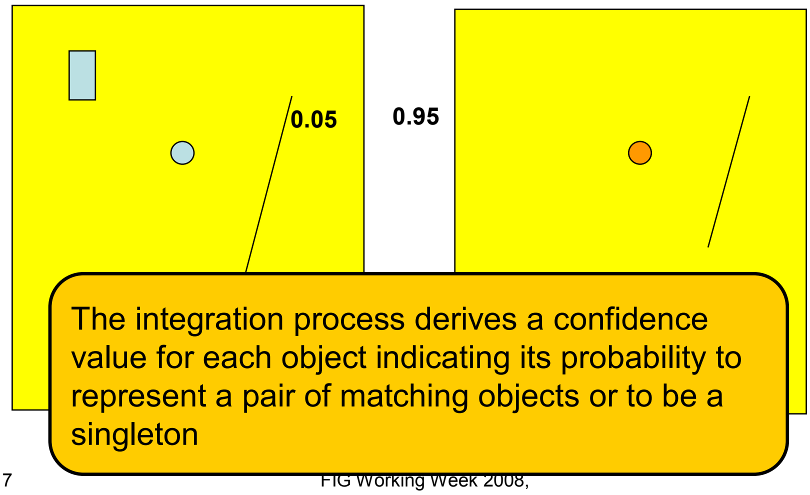
Is 100$ cheep?

Does the user mean the city of London or maybe he

A confidence value may be matched to any search result, indicating its probability to satisfy the user needs

6 FIG Working Week 2008, Stockholm, Sweden

Example – Datasets Integration



The Orienteering Problem (OP)

- When computing a route, different goals and constraints can be defined, such as minimizing the traveling length, limiting the route to be over roads of a certain type...
- We consider a route search where the aim is finding a route that starts at a given location and traverses through as many correct objects as possible without exceeding a given distance
- This problem is a generalization of the **Orienteering Problem (OP)**

Efficiency & Scalability

- Finding a solution to OP is a problem that cannot be computed efficiently
- For applications on the Web, it is crucial that an answer

Our main goal is to present heuristics to OP that are efficient and scalable. This will show that building a route-search system as a web application is realistic.

- with a bound as tight as possible

9 FIG Working Week 2008,
Stockholm, Sweden

OP Heuristics

- We present four efficient OP route-search heuristics
- We use a naive **Greedy** heuristic as a benchmark for measuring the performances of the other three heuristics
- Differently from the greedy approach, our heuristics lead the constructed route towards clusters of objects as soon as possible, because clusters allow collecting many objects that are likely to be correct, from a small area

10 FIG Working Week 2008,
Stockholm, Sweden

The Greedy Heuristic

At each step, the object having the minimal distance - confidence ratio is chosen

$$\frac{50}{0.70} = 71.43 \qquad \frac{60}{0.90} = 66.67$$

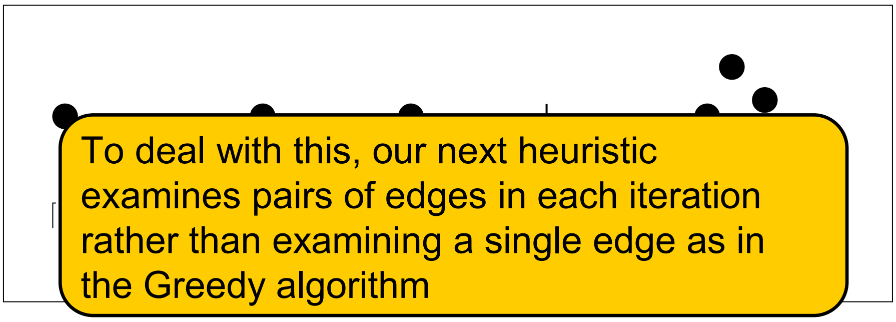
11 FIG Working Week 2008,
Stockholm, Sweden

The Greedy Heuristic

- It is simple and relatively efficient. No preprocessing is required and it has O(|*D*|^2) time complexity
- The greedy algorithm computes a good Orienteering route when the objects of *D* are uniformly distributed and their confidence values have a small variance, However it performs poorly in clustered datasets

12 FIG Working Week 2008,
Stockholm, Sweden

Example 4 – When The Greedy Heuristic Performs Poorly

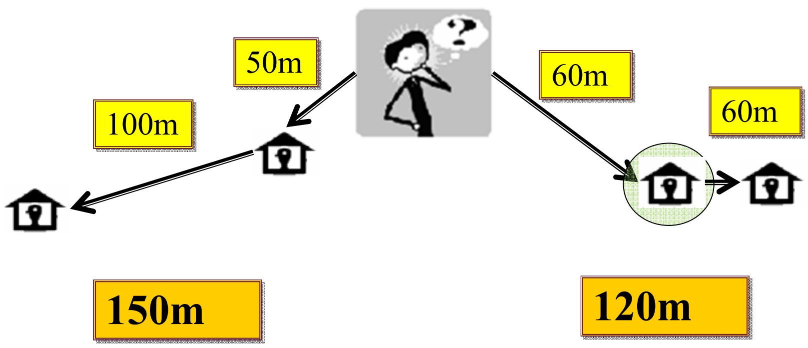


To deal with this, our next heuristic examines pairs of edges in each iteration rather than examining a single edge as in the Greedy algorithm

The Greedy algorithm creates a route from the objects that are on the left side with respect to the starting point, while missing the cluster on the right side

The DG Heuristic

At each step, the pair of objects having the sum of minimal distance - confidence ratio is chosen



The DG Heuristic

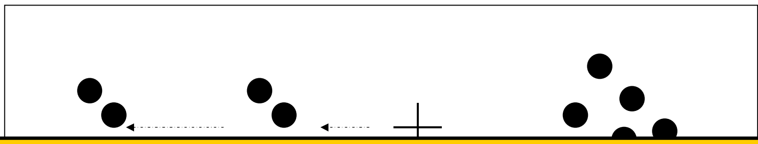
- The Double-Greedy Algorithm (DG) is an improvement of the Greedy Algorithm that, intuitively, examines pairs of edges for deciding which node to add
- Algorithm DG has time complexity O(|*D*|^3)
- In order to increase efficiency, DG checks a pair of edges only when the next edge we consider to add to the route is much longer than its preceding

The DG Heuristic

- Frequently, this indicates that the route is leaving a cluster

we want to detect this, and we want to direct the route to a new cluster

Example 5 – When The DG Heuristic Performs Poorly



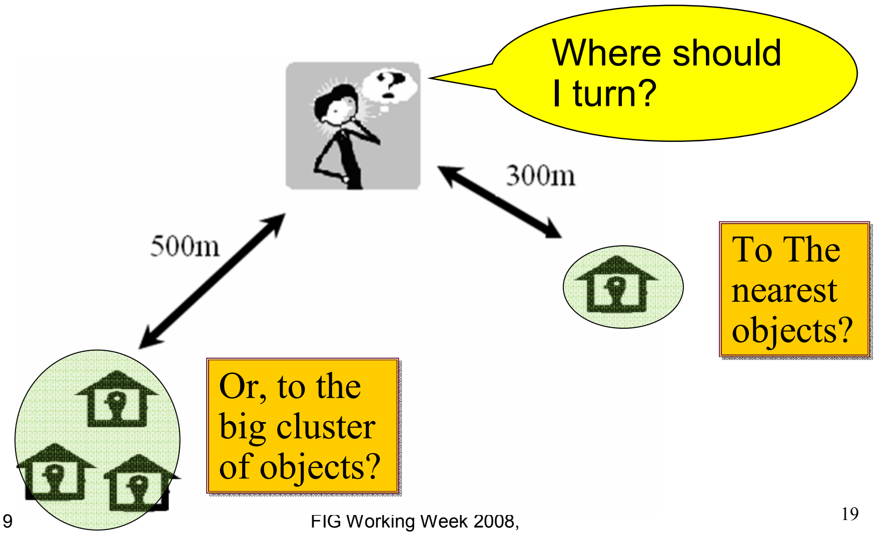
This motivates us to suggest additional heuristics. Their main idea is to give precedence to objects that are in a cluster over objects that are not in a cluster

In Former Greedy example , DG will return a route that goes to the cluster, however here it produces a route that turns away from the cluster on the right side. Thus, DG is not always a good heuristic

Handling Clustered Datasets

- Given a dataset that contains clusters of objects, a good heuristic for constructing an OP route is to give precedence to objects that are in a cluster over objects that are not in a cluster
- Clusters allow collecting many objects with little effort by which improving the Orienteering route result
- This approach is implemented by the AAG and AAGB heuristics

Objects spatial relationship



The AAG Heuristic

(Adjacency Aware Greedy)

- AAG starts by applying a **pre-processing step** in which the nodes confidence value is replaced

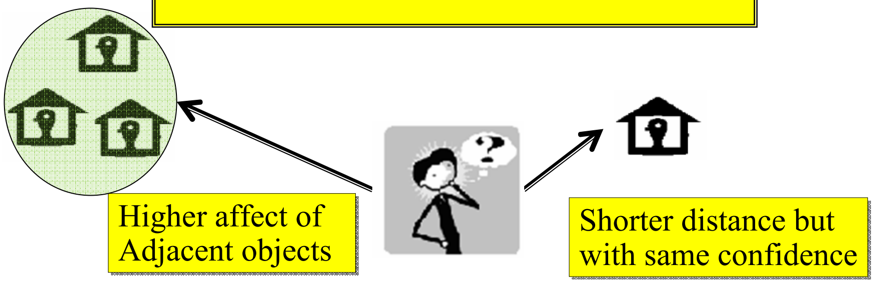
The AAG improves the Greedy algorithm by giving a higher weight to objects that have many near neighbors, especially if the near neighbors have high confidence values

of weights to compute its route

The AAG Heuristic

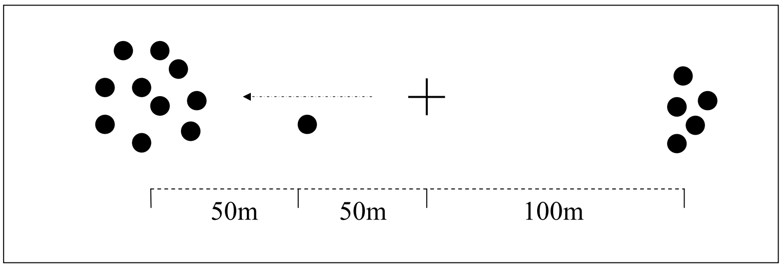
The new weights ***W*** takes into account:

- Object's **confidence** level
- Adjacent objects **proximity**



W value of the left group is higher do to the large number of Adjacent objects

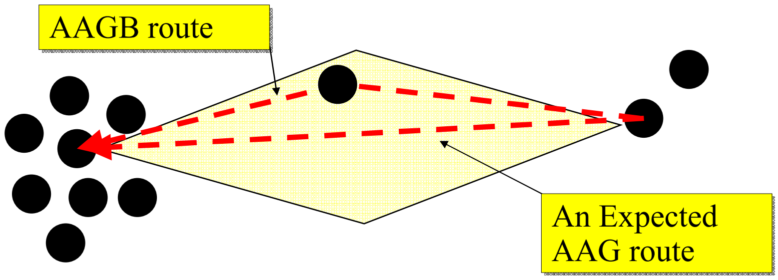
Example 6 – Motivation for Improving the AAG



The AAG turns to the bigger cluster on the left

Our goal is to improve AAG by including in the route objects that are near its route. This will increase the collected prize at a slight cost in length

The AAGB Heuristic



The AAGB collects near objects in the direction of route while perusing clusters the same as the AAG

The AAGB Heuristic

- We start by a similar computation as in AAG, but for each edge in the route, we build a buffer
- Objects that are inside the buffer are added to the route
- The shape of the buffer was chosen as a diamond in order to subtle coarse turns which may occur inside a buffer crossing

Comparing Results

- The Greedy heuristic is the most efficient and scalable among the four heuristics, but often it is less effective than the other methods, especially over datasets that have clusters

- The DG heuristic is an improvement of the Greedy, Yet both can miss nearby clusters when creating the route

Comparing Results

- Our tests show that AAG is more effective than DG and Greedy at the cost of a longer computation time

Our tests show that AAGB is the most effective heuristic but the least efficient among the four heuristics

- in a cluster but can be added to the route at a small cost

Further Work

- **Examine ways to include these methods in a comprehensive route-search system**
- **Adding complex constraints to the route search**
  - Multiple entity search (i.e. 5 hotel and 4 restaurants)
  - Incorporating order constraints (i.e. a gas station after the fourth tourist attraction)

Questions?