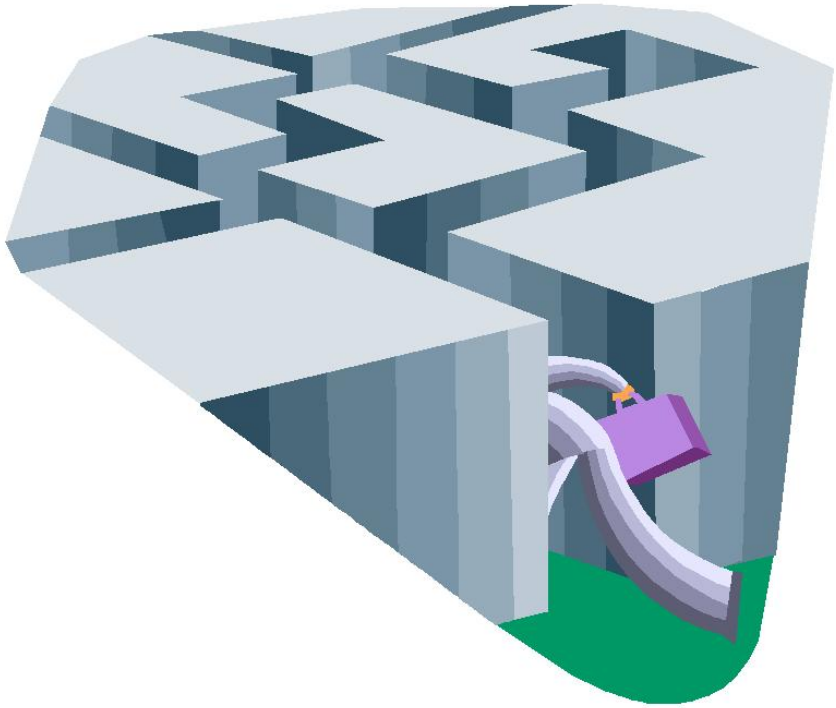
Algorithms for optimization under uncertainty

Universal and a priori algorithms for the

Traveling Salesman Problem (TSP)



Universal and a priori Traveling Salesman Problem

Traveling Salesman problem

Given n points in a square find the shortest tour passing through each city and returning to the initial point.

Distance= Euclidean distance

Problem is NP hard

Also: points in the plane → nodes of graph with weight on the edges.

Many approximation algorithms:
A simple 2-approximate solution is based on Minimum Spanning Tree.

A different scenario:

Initially we are given a set N of n points and we are asked to find a tour T through this set of points.

Then we are given each day a subset A of points A subset of N.

Given T and A it is easy to find a tour through A by shortcutting T to points in A

Universal Algorithm
Find a tour T s.t. for all A shortcutting T gives a tour for A that is closest o the optimal tour through A.

Motivations

- ❖speed: given the initial tour it is trivial to compute the daily tour;
- ❖each day drivers have a similar tour even if the points change

Space filling curves

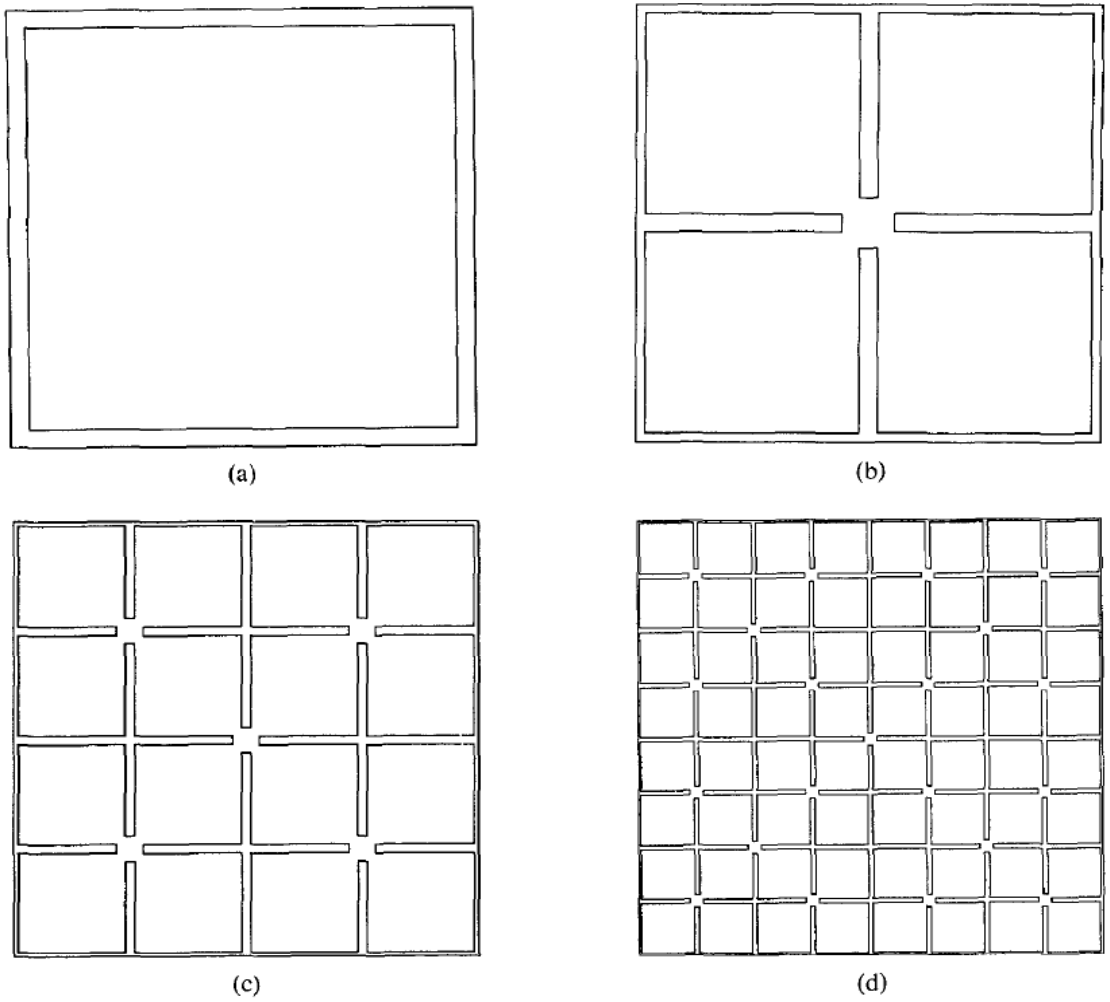


FIG. 3. The conventional representation of an approximation to a spacefilling curve shows a path through the marked vertices of the triangles of Figure 1, sequenced according to their labels. (a) $k = 2$. (b) $k = 4$. (c) $k = 6$. (d) $k = 8$.

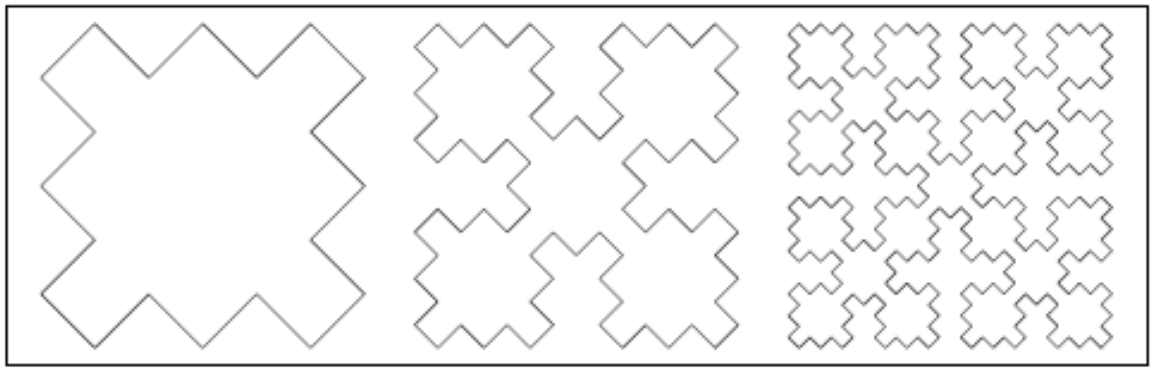


Figure 1: The Sierpinski spacefilling curve is the limit of the series of recursively-constructed figures shown above. Each is built upon the preceding figure by dividing the square into quadrants and filling each quadrant with a shrunken copy of the preceding figure. The limiting figure is a 1-dimensional curve that is continuous and visits every point in the 2-dimensional square.

Parameter: number of iterations # iter

Intuition: as #iter increases then max distance of a point in the square from the space filling curve diminishes by a constant factor each time

Properties If #iter goes to infinity cover all space; continous

ALGORITHM

Step 1. For each location p= (x, y) calculate its relative position position(p) along the spacefilling curve (i.e. the point on the curve that is closest to p); position(p) is a number between 0 and 1.

Step 2. Sort the locations from the smallest to largest position(p)

Thus the locations to be visited are sequenced according to the order in which they appear along the spacefilling curve, as illustrated in Figure

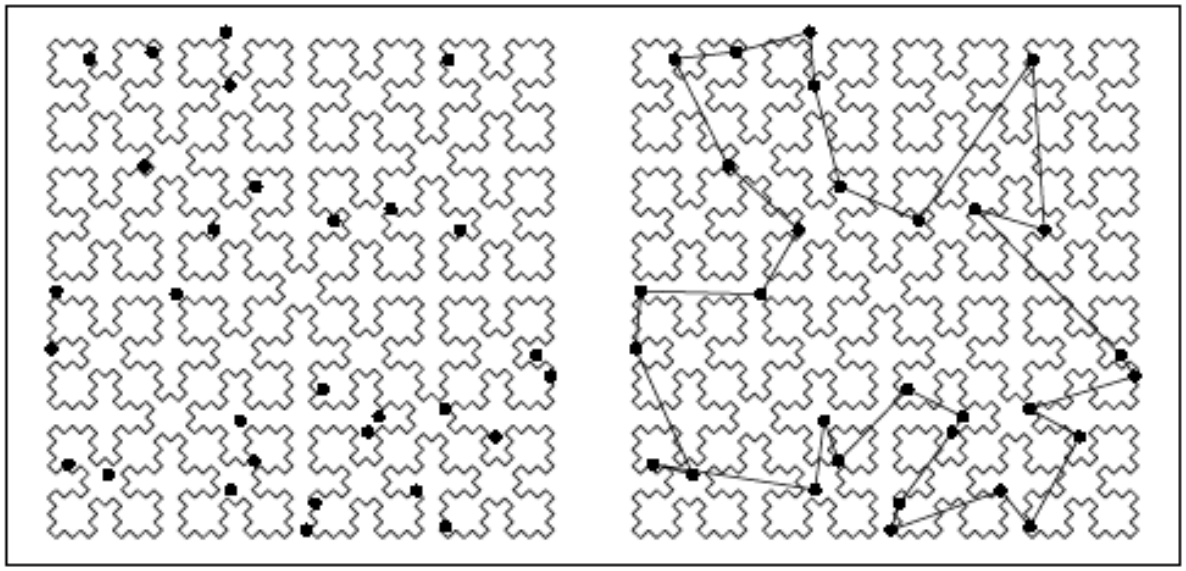


Figure 2: A set of random points and a TSP tour according to their relative position along the spacefilling curve.

Implementation

Given a point compute its position on the curve.

Assume southwest corner is the beginning and that L is the total length of the curve.

Fix a direction of the curve – say clockwise

Given a point p let D(p) be the distance of the point from the origin

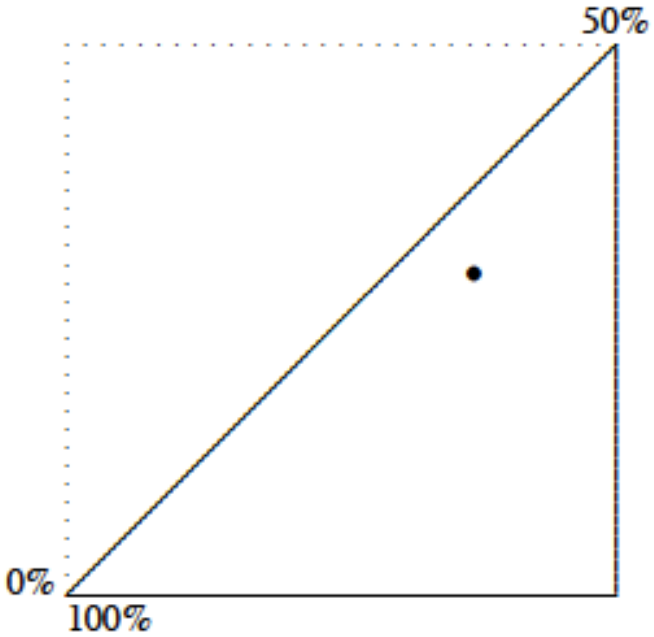
Position(p) = D(p) / L.

So the origin is represented by 0 (beginning) and 1 (end)

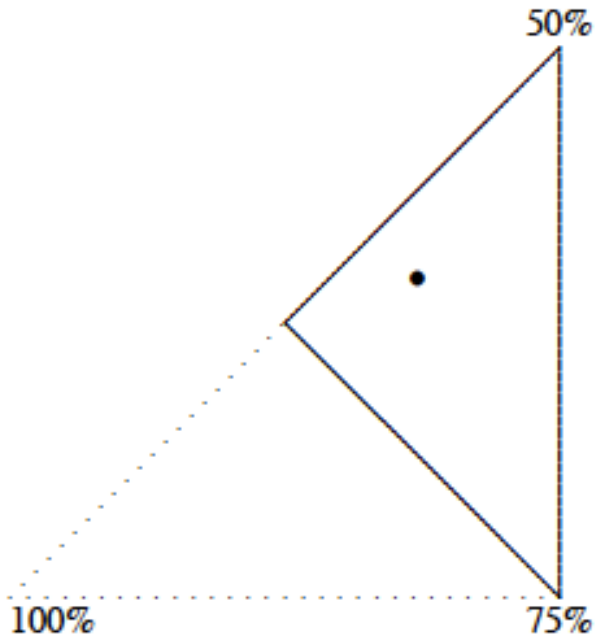
To find the position of a point we proceed by successive refinements: at each step divide the region containing the point in two triangles

And compute in which subtriangle the point is

Step 1: From Figure 1 you will notice that the Sierpinski curve visits all points in the northwest triangle before visiting any points in the southeast triangle. This means the northeast corner must be 50% of the way around the curve; and so our point p, which lies within the southeast triangle, must be 50–100% of the way around the curve (Figure 3).

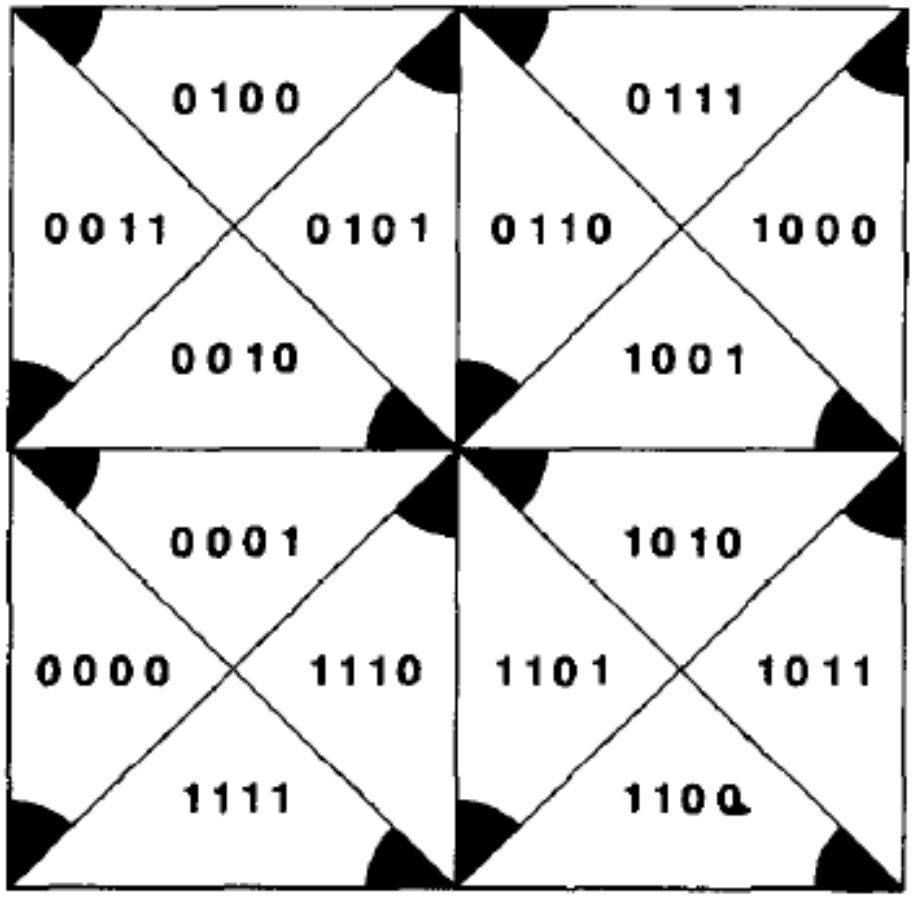


Our point is in the interval [0.5, 1]



Our point is in the interval [0.5, .75]

We continue in this way. Let d be the precision parameter



(d)

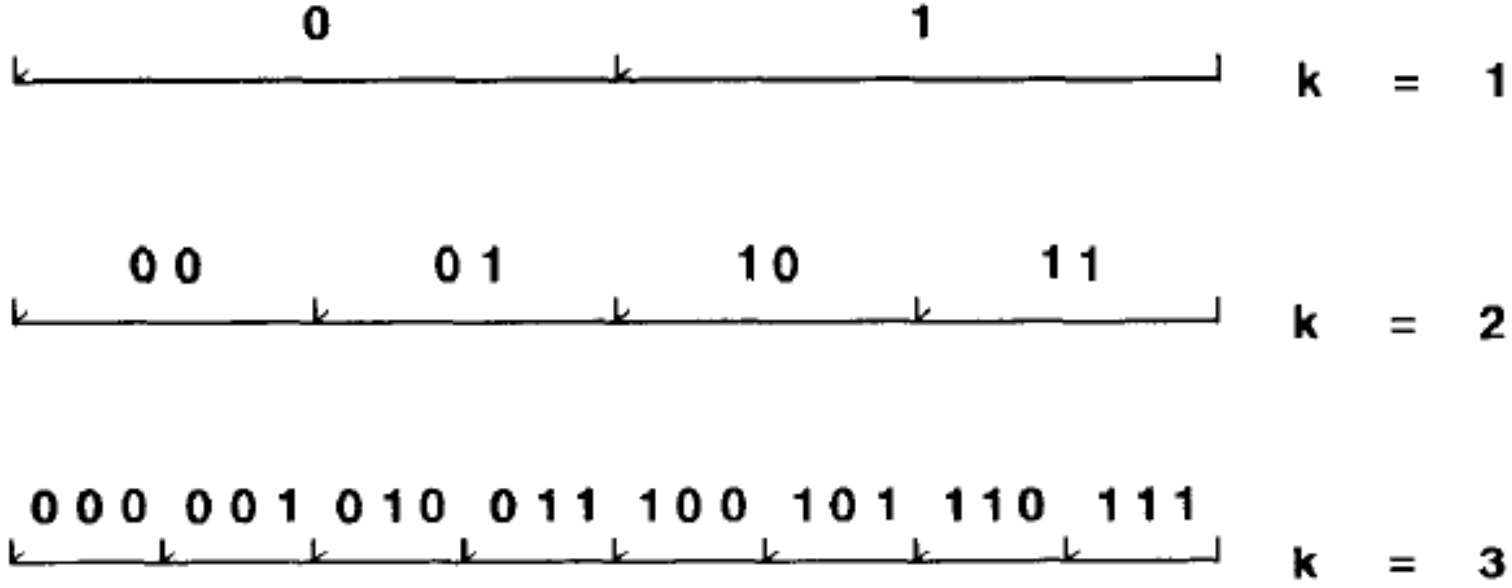


FIG. 2. Successive partitions of the unit interval C into 2^k identical subintervals. The label contains the first k binary digits of numbers in the subinterval's interior. The marked endpoint is always to the left.

Analysis of the space filling heuristics

Two initial lemmas

The first Lemma formalizes an observation of [9, p. 65] that a spacefilling curve preserves nearness: points close together in C map (via ψ) onto points close together in S. We take the measure of nearness on the square S to be Euclidean distance, denoted by $D[\cdot, \cdot]$. Since we consider C to be a circuit with $\psi(0) = \psi(1)$, the natural metric on C is

$$\Delta[\theta, \theta'] = \min\{|\theta - \theta'|, 1 - |\theta - \theta'|\}.$$

Using these metrics, we can now state

LEMMA 2.1. $D[\psi(\theta), \psi(\theta')] \leq 2\sqrt{\Delta[\theta, \theta']}$ *for all* $\theta, \theta' \in C$.

Lemma 2 : Space filling are Borel measurable (so we can perform integrals) and they cover uniformly the space (i.e. a segment x "covers" a region Area(x)= lenght (x))

Given a set of points A, let

L* = length of optimal tour L= length of tour found by algorithm

Π = Set of given points

THEOREM 4.2. $L/L^* = O(\log N)$.

PROOF. Let $\#\{\cdot\}$ denote cardinality, and $1\{\cdot\}$ the indicator function. If w_k is the length of the kth link along the tour, and

$$H(t) = \#\{k : w_k \geq t\},$$

then

$$L = \sum w_k = \sum \int_0^\infty 1\{w_k > t\}\, dt = \int_0^\infty H(t)\, dt. \tag{4.1}$$

We show that there is a number Ω (independent of Π) such that

$$H(t) \leq \frac{\Omega L^*}{t}, \qquad \text{for all} \quad t \in [0, L^*]. \tag{4.2}$$

Since $H(t) \leq N$ for all t and $H(t) = 0$ for $t > L^*$, (4.1) and (4.2) combine to give the desired result,

$$\frac{L}{L^*} \leq \int_0^{L^*} \min\left(N, \frac{\Omega L^*}{t}\right) \frac{dt}{L^*} \leq \Omega \int_0^1 \min\left(N, \frac{1}{a}\right) da = O(\log N).$$

a

To obtain (4.2), consider the set $\Pi(t)$ of points in S that lie within distance t of some point in Π. Also, let Π' denote a collection of $\lceil L^*/t \rceil$ points placed at equal intervals along the path of the optimal tour, so that each point in Π lies within distance t of some point in Π'. Now each point in $\Pi(t)$ must lie within distance $2t$ of a point in Π', so

$$\text{area}\{\Pi(t)\} \leq \left\lceil \frac{L^*}{t} \right\rceil \pi(2t)^2 \leq 8\pi t L^*, \; t \in [0, L^*]. \tag{4.3}$$

Finally, let $[\theta_k = \lambda(p_k), k = 1, \ldots, N]$ be the sorted list from which the heuristic tour is constructed, and consider the intervals

$$C_k(t) = \left\{ \theta \bmod 1 : \theta_k < \theta < \theta_k + \frac{t^2}{4} \right\}.$$

By Lemma 2.1, if $w_k \geq t$, then $C_k(t) \cap C_{k'} = \varnothing$, $k \neq k'$. Thus, there are at least $H(t)$ mutually disjoint intervals $C_k(t)$. Moreover, by Lemma 2.1, $\psi(C_k(t)) \subseteq \Pi(t)$, and by Lemma 2.2, $\psi(C_k(t))$ has area $t^2/4$. It follows from Lemma 2.2 that

$$\text{area}\{\Pi(t)\} \geq H(t) \frac{t^2}{4}. \tag{4.4}$$

Note

The result is independent of the chosen space filling curve (the choice affects only the constant factor)

Other results

There is worst cases matching the bound

No algorithm can achieve a ratio better than $(\log n / \log \log n)^{(1/6)}$ Hajaghavi et al.

If points are unformaly distributed in the plane then the algortithm provides a constant approximation (almost always).

QUESTIONS?

"More decisive? How can I be more decisive?
- I live by the uncertainty principle!"

A priori TSP

Input a set N of n points and a vector P; point i can be ACTIVE with probability p(j)

Let A be the set of current active points; we are interested in finding a tour through points in A.

Goal: find one tour T through all n points s.t. when A is given shortcutting T and passing through only points in A we obtain a good TSP tour through A.

For the *a priori* TSP, the input consists of a set of points N, a distance function d, and a probability distribution Π specified over the subsets of N. Since we are working in the independent activation model, we can compute the probability $\Pi(A)$ for each subset $A \subseteq N$; that is,

$$\Pi(A) = \left(\prod_{j \in A} p_j \right) \left(\prod_{j \notin A} (1 - p_j) \right), \qquad (1)$$

where p_j is the probability that point j is in the active set A. The goal now

Let T(A) be the lenght of the short cut tour through points in A; T*(A) be the optimal tour

Recall: we assume that the probability of occurrence of points is known.

THEOREM: If distribution of points is known then there exists a randomized algorithm such that $Exp[T(A)] < 8\ T^*(A)$

Proof : special case $p(1)=1$ (i.e. point 1 is always present in A).

Algorithm

1. Randomly Choose a set S of points according to the distribution of points .
2. Find a Minimum Spanning tree through S
3. Extend the Spanning tree to all points in N as follows: if p is not in S then add the edge from p the nearest point in S; let T be the final spanng tree
4. Construct a tour by doubling each edge of T and "walking around" T

Analysis

Let MST(S) be the lenght of the minimum spanning tree through S

Let D(j,S) be the minimum distance of j from a point in S

Given a set of A of active points we compute the lenght of the short cut tour

FACT1 $OPT(A) > MST(A)$ therefore $OPT(A) > Exp[MST(A)]$

FACT 2 $OPT(A) > \Sigma_{(j \text{ in } A)} D(j,A)$ therefore $OPT(A) > Exp[\Sigma_{(j \text{ in } A)} D(j,A)]$

Let T(A) be the spanning tree obtained from T by "shortcutting" points NOT in A let L(T(A)) be its lenght

We now show that $Exp[L(T(A))] < 4\, OPT(A)$

It follows that, by walking around through T(A), we obtain a tour of legnth $< 2\, MST(T(A))$ we obtain the thesis.

We have

$L(T(A)) < MST(S) + \Sigma_{(j \text{ in } A),\ j \neq 1} D(j,S)$

Note that T(A) depends on two random variables S and A

(Exp_S denotes expected value over random variable S;

Ind(X) indicator function that is 1 if event X holds, 0 otherwise)

$Exp_S [Exp_A[L(T(A))]]$

$< Exp_S[Exp_A[MST(S)]] + Exp_S[Exp_A[\Sigma_{j \neq 1} Ind(j \text{ in } A)\ D(j,S)]]$

$= Exp_S [MST(S)] + Exp_S [Exp_A [\Sigma_{j \neq 1} Ind(j \text{ in } A)\ D(j,S)]$

$Exp\_S\ [Exp\_A[L(T(A))]] <\ Exp\_S\ [MST(S)]\ + Exp\_S\ [Exp\_A\ [\Sigma_{j\neq 1}\ Ind(j\ in\ A)\ D(j,S)]]$
FACT1 $OPT(A) > MST(A)$, $OPT\ (A) > Exp[MST(A)]$
FACT 2 $OPT(A)\ > \Sigma_{j\neq 1}\ Ind(j\ in\ A)\ D(j,A)$, $OPT\ (A) > Exp\ [\Sigma_{j\neq 1}\ Ind(j\ in\ A)\ D(j,A)]$

Since S is chosen according the same prob. distribution as A we have

$Exp[MST(S)]= Exp[MST(A)] < OPT$

Since S and A are chosen according to the same distribution and are independent we have

$$Exp\_S\ [Exp\_A\ [\Sigma_{j\neq 1}\ Ind(j\ in\ A)\ D(j,S)]] = \Sigma_{j\neq 1}\ Exp\_A\ [Ind(j\ in\ A)]\ Exp\_S\ [\ D(j,S)]$$

$$= \Sigma_{j\neq 1}\ Exp\_A\ [Ind(j\ in\ A)]\ Exp\_A\ [\ D(j,A)]$$

Now the distance of j is independent of the fact that j is in A. So

$$\Sigma_{j\neq 1}\ Exp\_A\ [\ Ind(j\ in\ A)]\ Exp\_A\ [\ D(j,A)] = \Sigma_{j\neq 1}\ Exp\_A\ [Ind(j\ in\ A)\ D(j,A)] <\ OPT$$

It follows that $Exp\_S\ [Exp\_A[L(T(A))]] < (OPT + OPT) = 2OPT$

NOTE

The algorithm is randomized

There exists (more complicated) algorithms that hold also when distances are NOT Euclidean