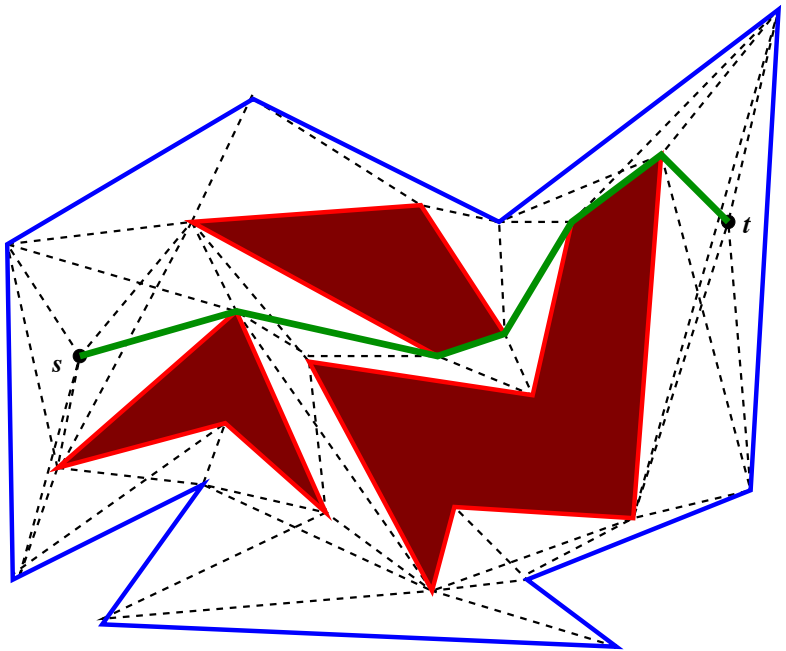
Optimal Tours, Paths and Networks in Geometric Settings: Some Recent Results and Continuing Challenges

Joe Mitchell

State University of New York
Stony Brook, NY 11794–3600

Example Problem 1:

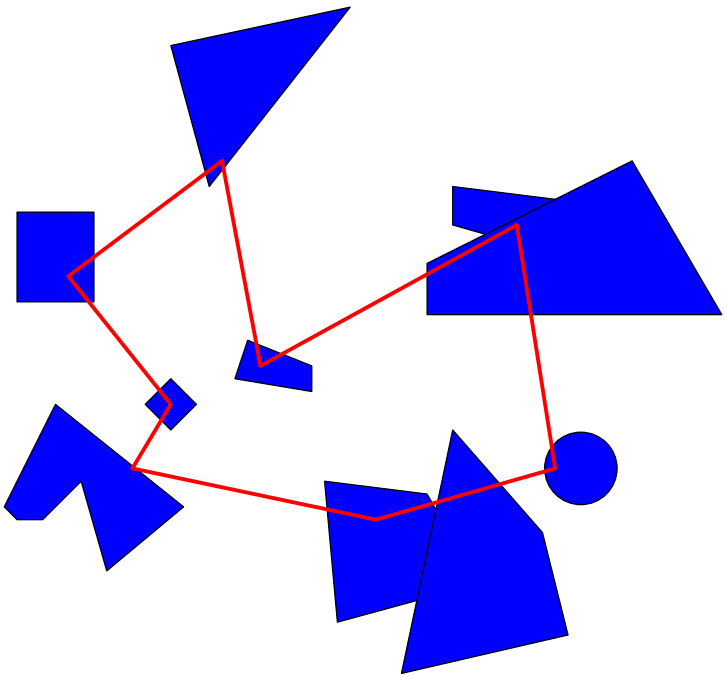
Shortest Paths Among Obstacles: Visibility Graph



Air Traffic Management: Multiple "well separated" routes among moving obstacles

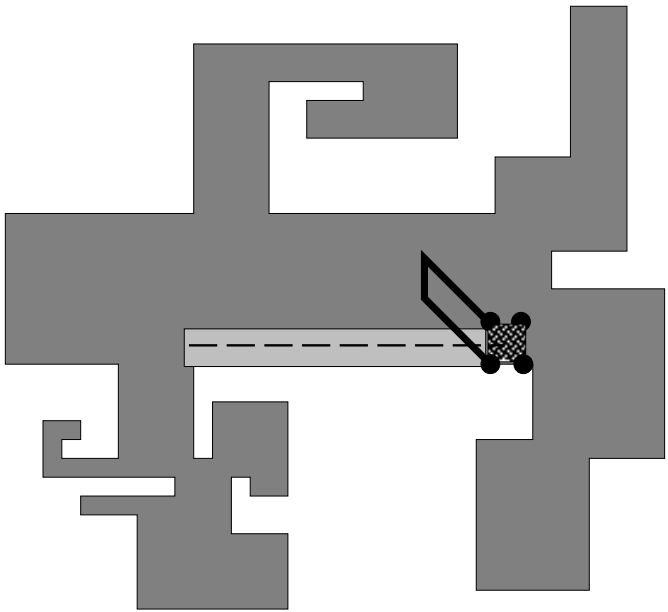
Example Problem 2:

Visit a set (unordered) or a sequence (ordered) of cities/regions



Example Problem 3:

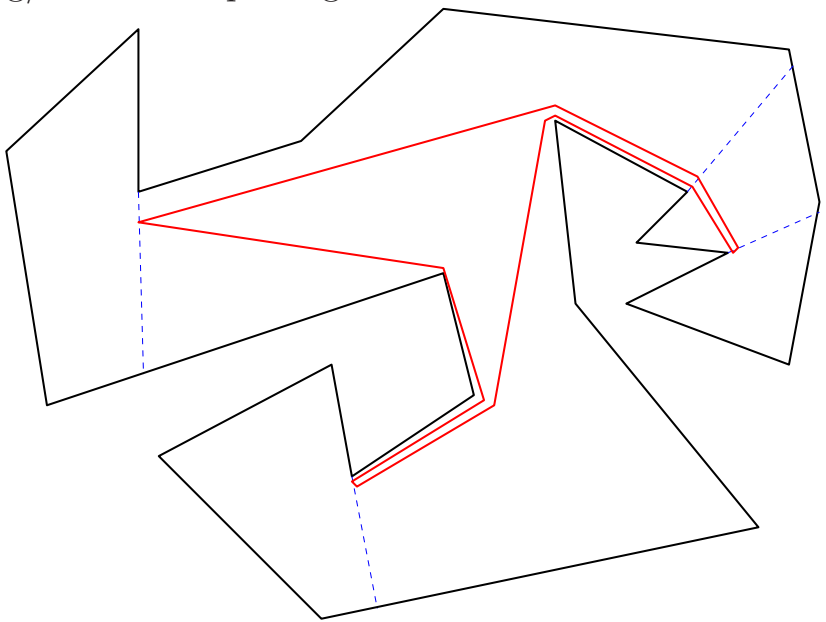
Optimally mow a lawn or mill a pocket



Worry about removing material: "Snow blower problem"

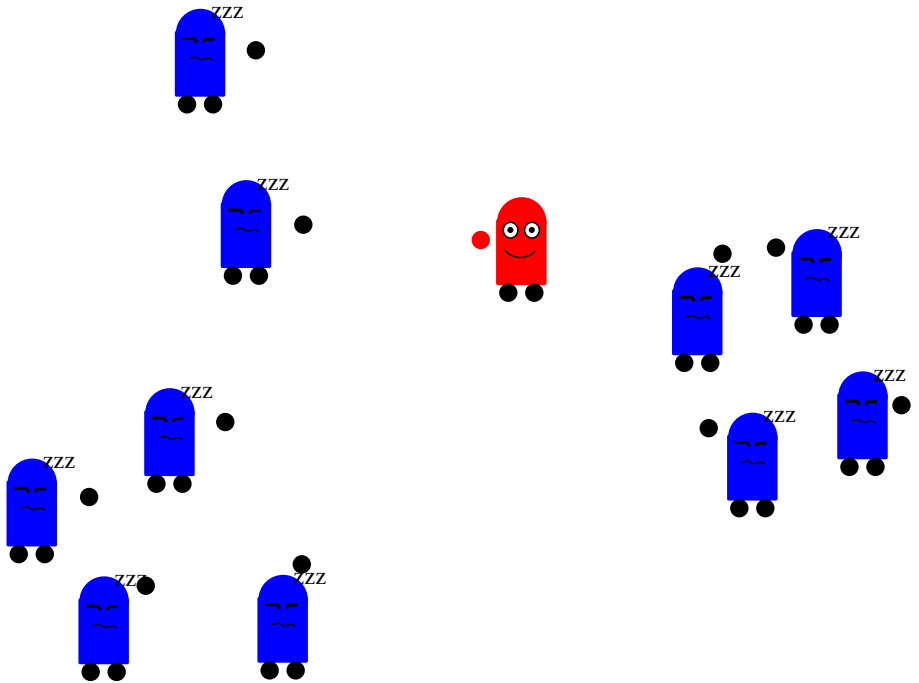
Example Problem 4:

Search a building/view or map a region: Watchman Route



Example Problem 5:

Awaken a swarm of robots: "Freeze-Tag" Problem



General Geometric Optimal Tour Problem:

Given a geometric domain $\mathcal{D} \subset \Re^d$, find an "optimal" tour/path in $\mathcal{D}$ that satisfies a given set of constraints:

- visit each element of a set S of points, lines, regions, etc.
- "see" all of $\mathcal{D}$
- mill/mow/sweep the domain $\mathcal{D}$
- pass through a given *root* point

" **Optimal**" can have various meanings:

- shortest total Euclidean length
- shortest total L_p length
- minimum number of links (link metric)
- minimum amount of turning
- min the max edge length
- max the min edge length
- min/max area of the enclosed region
- etc.

Approximation Algorithms:

c-approximation: cost at most c times optimal,
for a minimization problem ($c > 1$)

Polynomial Time Approximation Scheme (PTAS):
method giving $(1 + \epsilon)$-approx to the optimal (minimum),
in time polynomial in n, for *any* fixed $\epsilon > 0$.

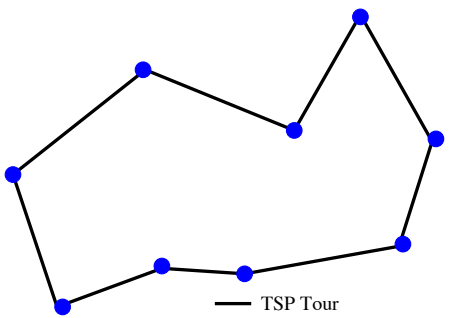
Dependence on ϵ may be exponential in $(1/\epsilon)$; else FPTAS

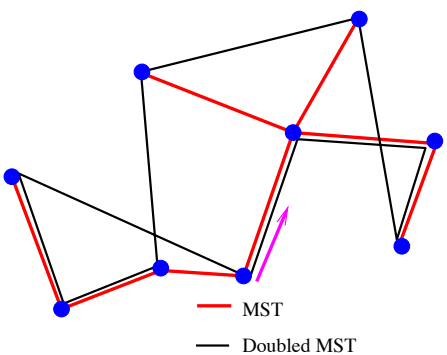
Overview of Talk:

- Approximation algorithms: Classical TSP
- Visiting a sequence of regions: Touring Polygons Problem (TPP)
- 3D shortest path problems
  - Some "simple" 3D shortest path problems that are hard
  - Poly-time solutions to some other "simple" 3D path problems
- A TSP variant: TSP with Neighborhoods
- Min-diameter bounded degree spanning trees: Freeze-Tag Problem

Classical TSP on Points:

- S = set of n points in $\Re^d$
- NP-hard
- $n^{O(n^{1-1/d})}$ exact (subexponential) [SmWo98]
- Simple 2-approx: double the MST and shortcut (holds in metric spaces)





- Christofides: 1.5-approx
  (use MST $\cup$ min-weight matching on odd-degree nodes of MST)

A Recipe for Approximation Algorithms:

- Show how to convert OPT to a special class $\mathcal{C}$ of solution instances

  Give THM showing that this conversion does not cost much [factor c_1]

$$L_{\mathcal{C}}^{\mathrm{OPT}} \leq c_1 \cdot L^*$$

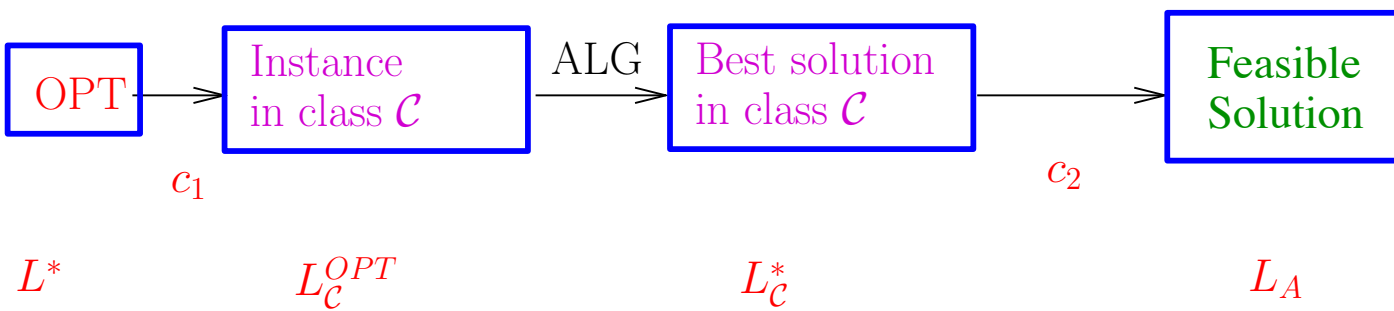
- Give an efficient algorithm for exact optimization over class $\mathcal{C}$
- Show how to recover a feasible solution to the original problem at small cost [factor c_2]

$$L_A \leq c_2 \cdot L_{\mathcal{C}}^*$$

A Recipe for Approximation Algorithms (cont):

Overall, then the approx solution has cost

$$L_A \leq c_2 \cdot L^*_{\mathcal{C}} \leq c_2 \cdot L^{\mathrm{OPT}}_{\mathcal{C}} \leq c_2 \cdot c_1 \cdot L^*$$



PTAS Results for Classical TSP:

- $O(n^{O(1/\epsilon)})$ in $\Re^2$ [Ar96,Mi96]
- $O(n^{O(1)})$ in $\Re^2$ [Mi97]
- $O(n(\log n)^{(O(\frac{d}{\epsilon}))^{d-1}})$ expected ($O(n^{d+1}polylog)$ det.) [Ar97]
- $O(n \log n)$ deterministic [RaSm98]
  Idea: t-spanners and "t-banyons"
- NP-hard to get $(1+\epsilon)$-approx in $\Re^{O(\log n)}$, for some $\epsilon > 0$ [Tr97]
- MAX-SNP-hard in metric spaces
  No c-approx for $c < 129/128$ ($c < 41/40$, asym.) [PV99]

Main Idea of PTAS's:

Transform OPT into a near-opt network of special recursive structure that allows efficent optimization by dynamic programming

Simple PTAS in $\Re^2$:

Special class $\mathcal{C}$: "m-guillotine subdivisions"

Theorem: For any polygonal subdivision S (edges E) of length L, and any integer $m > 0$, there exists an m-guillotine subdivision, S_G (edges E_G) of length at most $(1 + \frac{\sqrt{2}}{m})L$, with $E \subseteq E_G$.

[Rectilinear subdivisions: factor $(1 + \frac{1}{m})$]

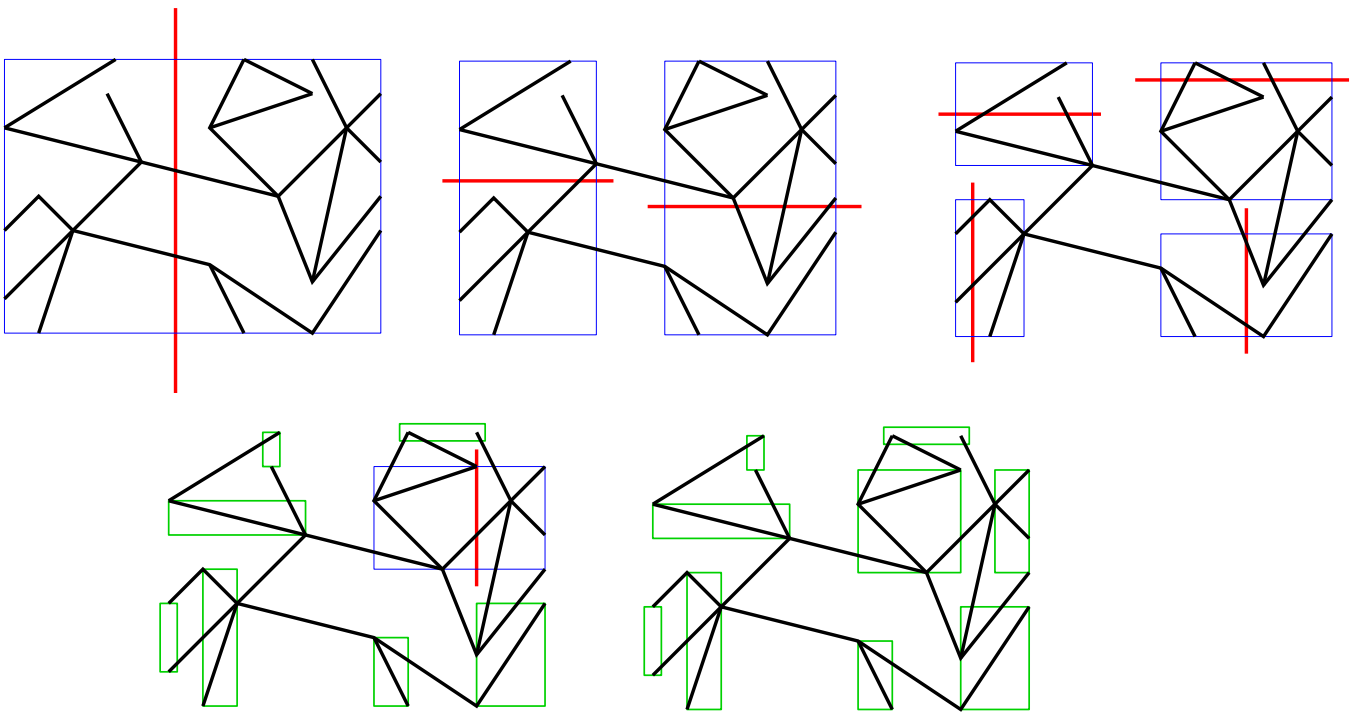
Algorithm: Optimizing over the class of m-guillotine subdivisions is easily done using dynamic programming (DP) algorithms.

Algorithm produces S_G^*, a *shortest* m-guillotine subdivision that obeys certain properties; e.g., connected, spanning all or some ($\geq k$) of the input points

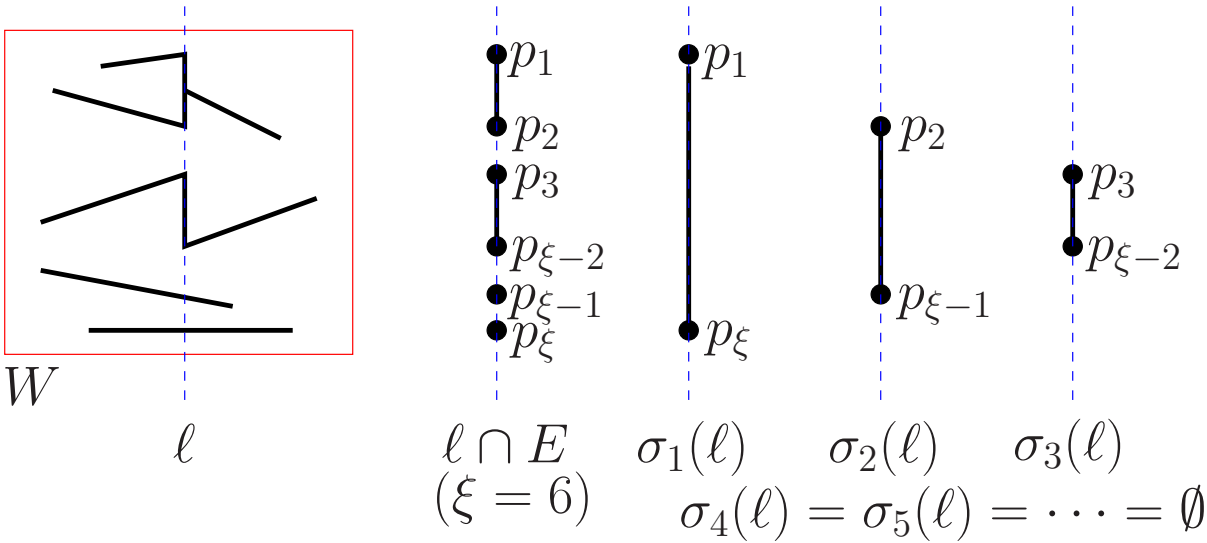
Solution Recovery from S_G^*:

- constraint in DP forcing Eulerian spanning subgraph of S_G^*

Example: 3-Guillotine Subdivision:



Definitions: m-span:

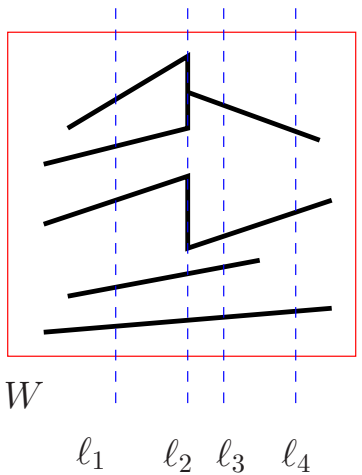


Definitions: m-perfect:

Example:

$\ell_1, \ell_2, \ell_3, \ell_4$ are 3-perfect (also m-perfect, $m \geq 4$)

ℓ_4 is also 2-perfect (but not 1-perfect)



Definitions: m-Guillotine:

Subdivision S is *m-guillotine* wrt W if either

(1) $V \cap int(W) = \emptyset$

OR

(2) $\exists$ m-perfect cut ℓ (wrt a minimal window, $\overline{W} \subseteq W$) such that S is m-guillotine wrt $W \cap H^+$ and $W \cap H^-$

(H^+, H^- are the closed halfplanes induced by ℓ)

Proof of Main Theorem:

Goal: Add new edges of total length $\leq (\frac{1}{m})L$ to make S m-guillotine

Recursive construction: At each stage, $\exists$ a cut, ℓ, such that we can add (if necessary) the m-span of ℓ to E, making ℓ m-perfect.

Charging scheme: For each unit of length we add (when we add $\sigma_m(\ell)$), there are m units of length of E to "pay" for it.

If m-perfect cut exists, use it, and recurse.
Thus, assume no m-perfect cut exists.

Proof of Main Theorem:

Consider making a horizontal cut, ℓ.

- *What does it "cost"?*

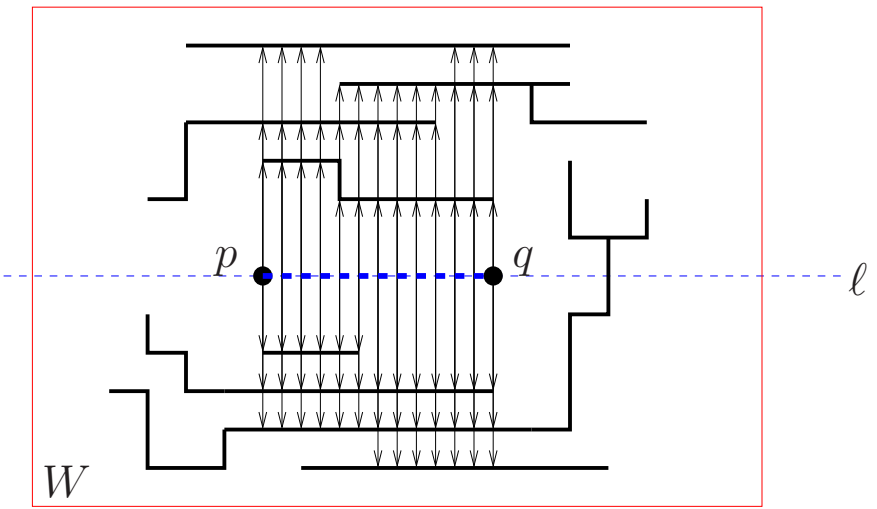
  Ans: We *add* (at most) the length of the m-span, $\sigma_m(\ell)$

- *Who can pay for it?*

  Ans: The "m-dark" points along ℓ

If horiz subseg $\overline{pq}$ (of length λ) is m-dark, we charge λ off to the bottoms of the first m subsegs above $\overline{pq}$, and the tops of the first m subsegs below $\overline{pq}$. Each level above/below "pays" $\frac{\lambda}{2m}$.

Example: Portion $\overline{pq} \subset \ell$ is 3-dark



Key Lemma:

Key Lemma: For any subdivision S and any window W, there exists a horiz/vert "favorable" cut whose m-dark portion is at least as long as its m-span.

For a favorable cut ℓ, we add its span to the edge set, and recurse on each side of ℓ.

Note: After a portion of E has been charged on one side, it will not be charged again on that side, since it will be within m levels of the boundary of any future windows.

Thus, no portion of E charged more than its length times $\frac{1}{2m} + \frac{1}{2m}$

Thus, the total length added to E is at most $(\frac{1}{m})L$.

Proof of Key Lemma:

Key Lemma: For any subdivision S and any window W, there exists a horiz/vert "favorable" cut whose m-dark portion is at least as long as its m-span.

- Let $f(x)$ = length of m-span of vertical line through x
  Let $g(y)$ = length of m-span of horizontal line through y

- Then,

$$A_x = \int f(x)dx$$

  is simply the area of the "m-dark" (RED) region wrt horiz cuts

  Similarly,

$$A_y = \int g(y)dy$$

  is the area of the "m-dark" (BLUE) region wrt vertical cuts

- Assume, WLOG, that $A_x \geq A_y$

- Thus, for $h(y) =$ length of m-dark, for horiz line through y,

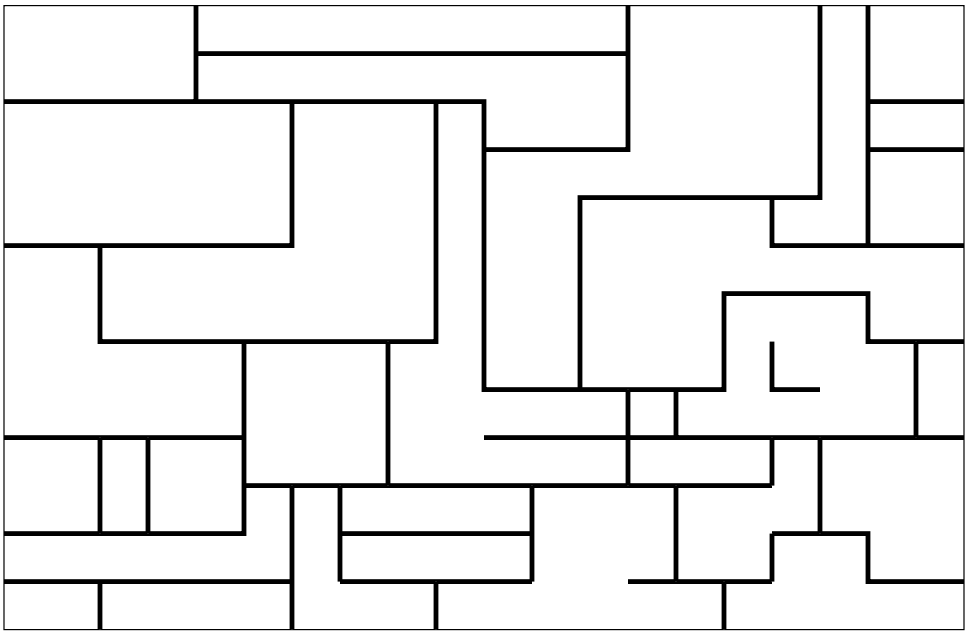
$$A_x = \int h(y)dy \geq \int g(y)dy = A_y > 0$$

So, $\exists y^*$ for which $h(y^*) \geq g(y^*)$;

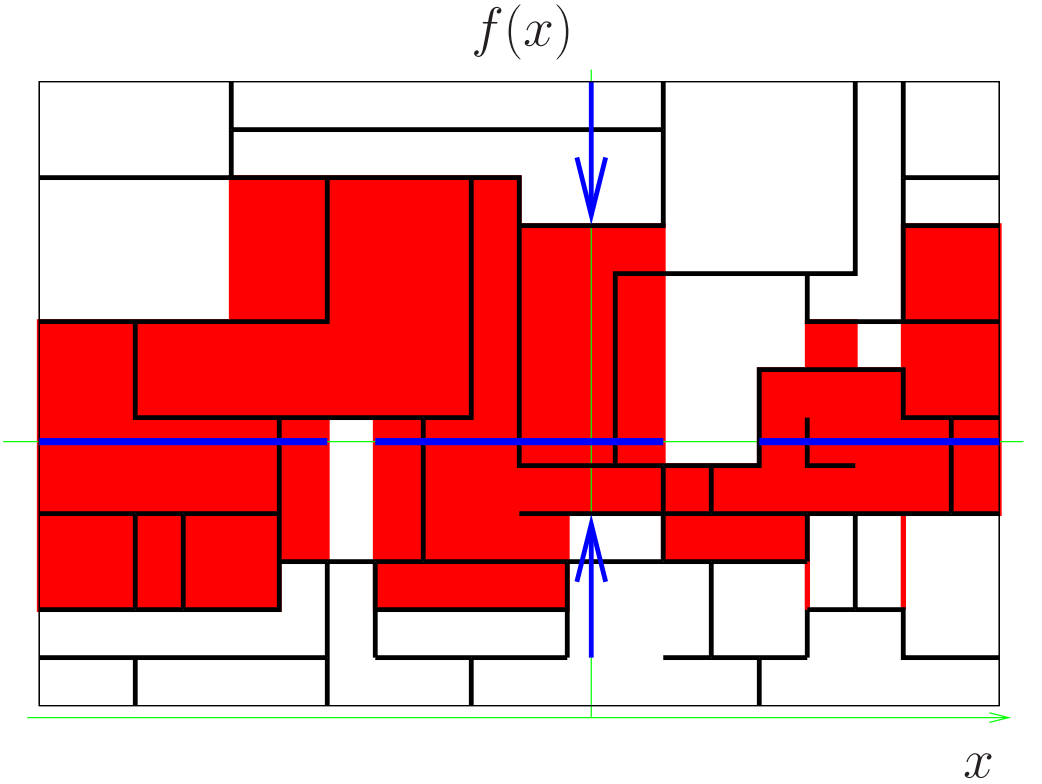
i.e., $\exists$ a horiz line through y^* whose m-dark portion $\geq m$-span.

(If $A_x \leq A_y$, then $\exists$ a vertical favorable cut.)

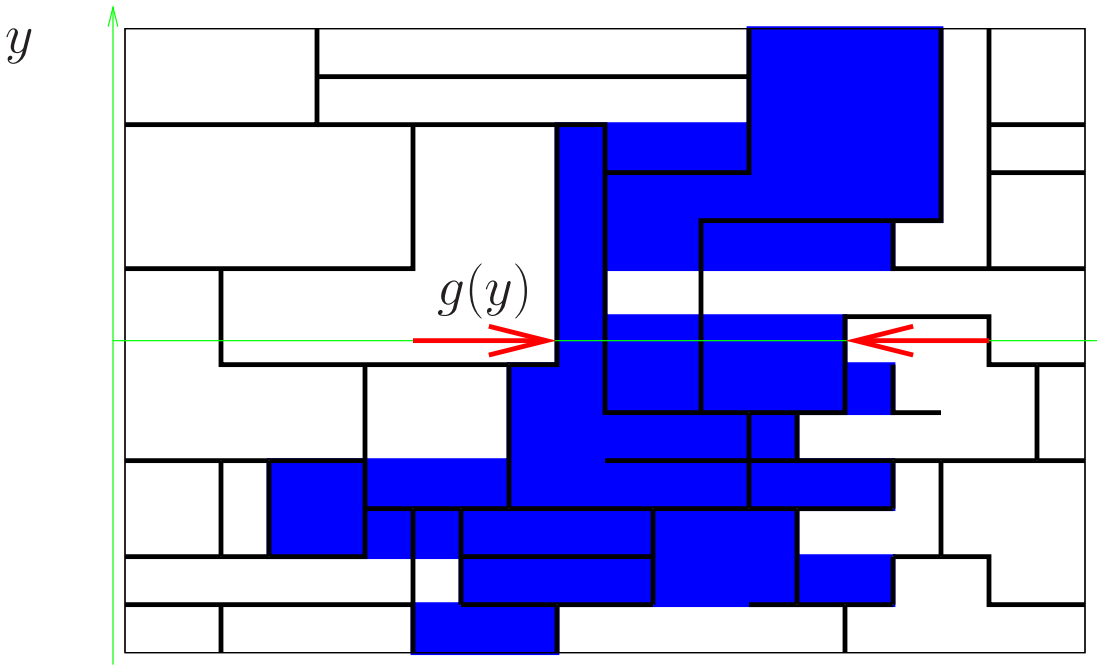
A rectilinear subdivision.



Region of 2-dark points wrt horizontal cuts (RED)



Region of 2-dark points wrt vertical cuts (BLUE)



Example Application: Steiner Tree:

Given: set P of n points (WLOG: no two have common x- or y-coord)

Consider the *rectilinear* (L_1) case first.

Let T_R = a min-length rectilinear Steiner tree for P; length L^*

By the Main Theorem, $\exists$ m-guillotine subdiv S_G, with edge set E_G, of length at most $(1 + \frac{1}{m})L^*$, such that $T_R \subseteq E_G$.

$\exists$ recursive cutting of S_G into windows, each side with $\leq 2m$ endpts

Thus, we can partition into *subproblems* having *small* descriptions.

Dynamic Programming Algorithm (Steiner):

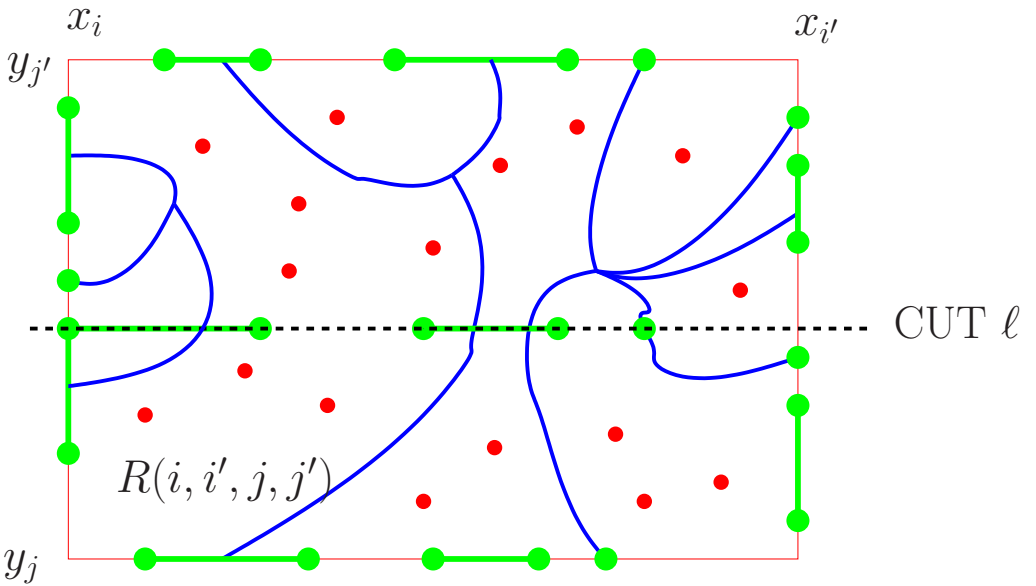
Sorted x-coord: $x_1 < x_2 < \cdots$ (for P, and grid lines)
Sorted y-coord: $y_1 < y_2 < \cdots$

Subproblem: $O(n^4 \cdot (n^{2m})^4) = O(n^{8m+4})$ choices

Input:

1. a rectangle $R(i, i', j, j')$, defined by x_i, $x_{i'}$, y_j, $y_{j'}$ $O(n^4)$
2. four sets of "boundary information", Σ_l, Σ_r, Σ_b, and Σ_t, determined by $\leq 2m$ endpoints on each side $O((n^{2m})^4)$
3. a partition, $\mathcal{P}$, of $\cup_\alpha \Sigma_\alpha$, giving required connectivity among boundary pieces $O(1)$

Objective: Find min-length m-guillotine subdivision, S_G^* (edges E_G^* interior to $R(i, i', j, j')$), such that E_G^* covers P and E_G^* connects the boundary pieces, according to partition $\mathcal{P}$.


x_i
$x_{i'}$
$y_{j'}$
CUT ℓ
$R(i, i', j, j')$
y_j

Dynamic Programming Algorithm (Steiner):

Let V = opt value of a subproblem
IF all connections required by $\mathcal{P}$ already
satisfied by boundary segments
THEN
$V = 0$
ELSE
Compute V recursively as the sum of the values of two subproblems obtained by splitting, minimized over:

1. $O(n)$ choices of a hor/vert cut
2. $O(n^{2m})$ choices of new boundary segments, Σ, on the cut
3. $O(1)$ choices of (compatible) partitions for resulting boundary segments in two new subproblems

Running time: — $O(n^{10m+5})$

Corollary: For any fixed positive integer m, there is an $O(n^{10m+5})$ algorithm to compute an approx Steiner tree whose L_1 length is within a factor $(1 + \frac{1}{m})$ of optimal.

Euclidean case: Same idea, but add a regular grid of potential (approx) Steiner points for P (spacing $\frac{diam(P)}{nM}$, for some $M \geq m$).

Corollary: For any fixed positive integer m, there is an $O(n^{O(m)})$ algorithm to compute a Steiner spanning tree (or Steiner k-MST) whose length is within a factor $(1 + \frac{1}{m})$ of minimum.