

The Snowblower Problem

Estie Arkin, Michael Bender,
Joseph Mitchell, Valentin Polishchuk

Stony Brook University

Suppose that your backyard looks like this:



One morning you wake up and it is covered with snow...



...uniformly covered



So you pull out your snowblower...



...or your **Snowblower**



...or your **SNOWBLOWER**



...and begin snowblowing

Depending on your backyard, snowblowing may look like this...



or snowblowing may look like this:



This talk: Algorithmic Aspects of Snowblowing

- Introduce the **snowblowing problem** (intermediate between TSP and material-handling problems)
- Give $O(1)$ approximation algorithms for several versions of the problem
- Prove NP-hardness for some versions of the problem

Snowblower: Material-Shifting Machine

- It lifts snow from one location, and piles it on an adjacent location



We must respect **max**
snow depth (height) D , because...



...if we pile snow up **too high**...



...the snowblower gets **stuck**



Where to **dispose of the snow?**

Where to **dispose of the snow?** (neighbor's yard)



We can pile snow arbitrarily high on the neighbor's lawn...



Effectively the neighbor's lawn
has infinite capacity



Alternatively: boundary is "cliff"
We dump as much snow as we want



(A bigger cliff)



We can achieve infinite capacity using a **snow melter**...



Snow Melter



SNOW MELTER



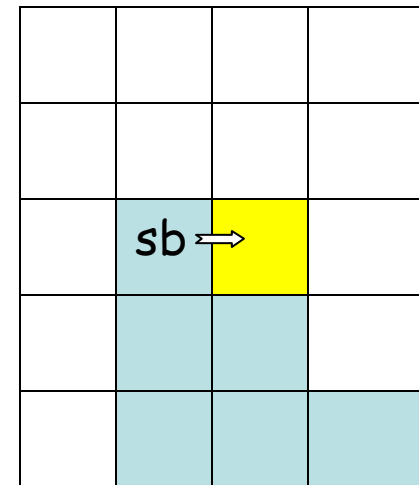
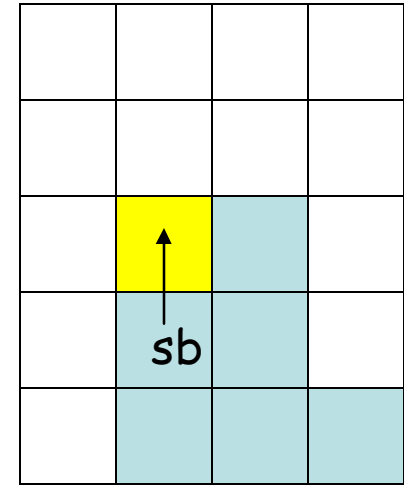
SB Problem Definition - Driveway

- Polygonal domain P
 - integral-orthogonal and pixelated
 - no holes
- SB
 - 1 pixel (initially in garage)



SB Problem Definition

- SB moves from pixel to adjacent pixel
 - picks up all snow
 - throws onto a neighbor pixel
 - or over the boundary of region
 - max depth of snow $D \geq 2$
- **Objective: minimize the length of the path of the snowblower**



The SBP is TSP-like

- **Milling/lawnmowing** [Arkin,Fekete,Mitchell00, ArkinHeldSmith00, Held91]
 - visit = remove
 - never re-visit \Rightarrow in NP
- **Material handling** [pushing blocks; extensive OR literature]
 - visit = move
 - may need to re-visit a lot \Rightarrow in NP?
- **The Snowblower Problem (*SBP*)**
 - visit = move
 - stacking $\leq D$ allowed
 - visit a boundary pixel = remove
 - our algs \Rightarrow in NP

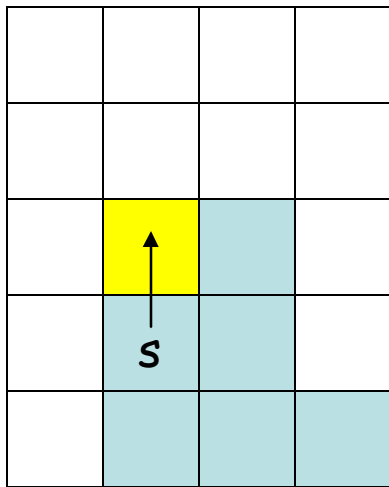
We are not the first to consider pixel environments for snowblowing...



City of Danville
Public Works Department
Danville, VA 24540

Throw Direction

On which pixel can snow be placed?



Right



Left



Forward?



Yes

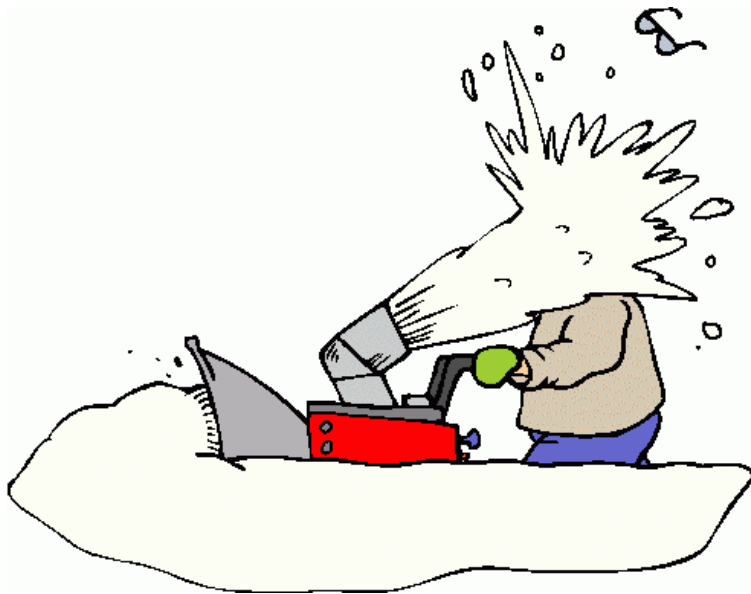
- if adjustable



Backward?

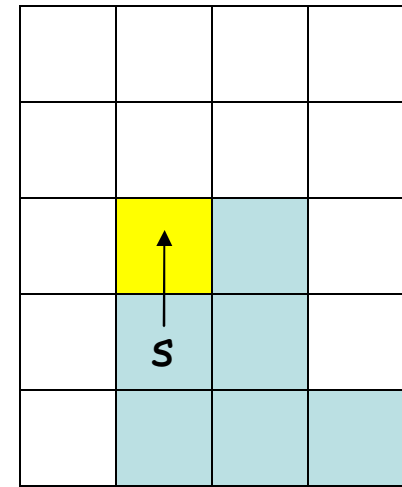
Backward?

- Not easy to implement. But it makes the algorithms easier.

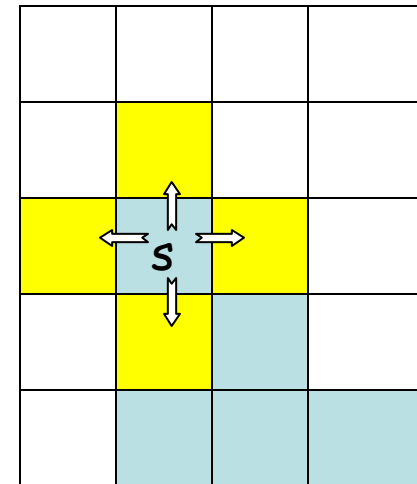


Default Model

- Throwback allowed
- Not intended to be realistic, but easy to describe and other models reduce to it



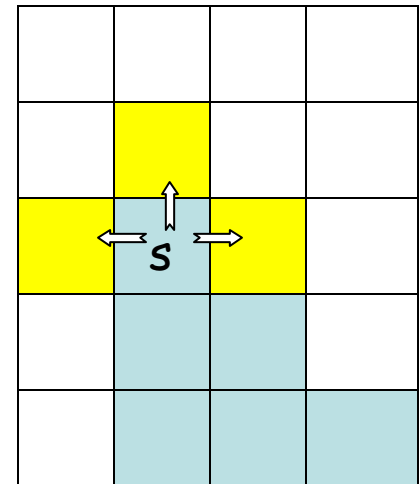
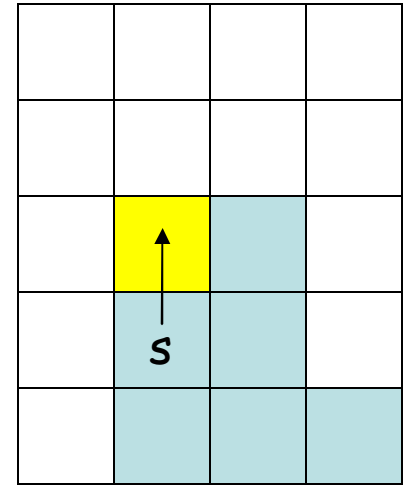
8-APX



Adjustable-Throw Model

Right, left, or fwd

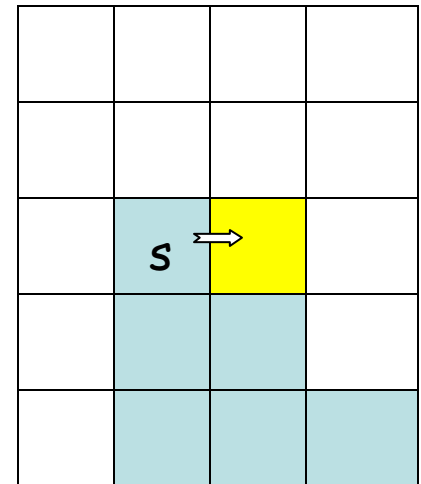
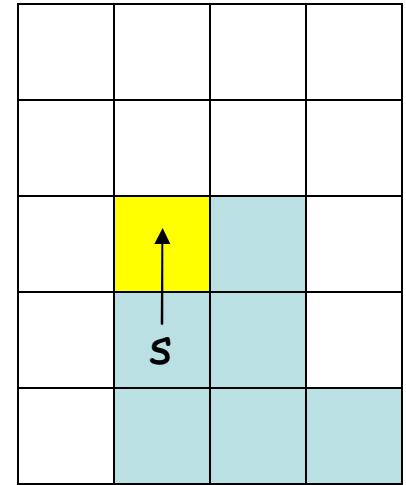
9-APX



Fixed-Throw Model

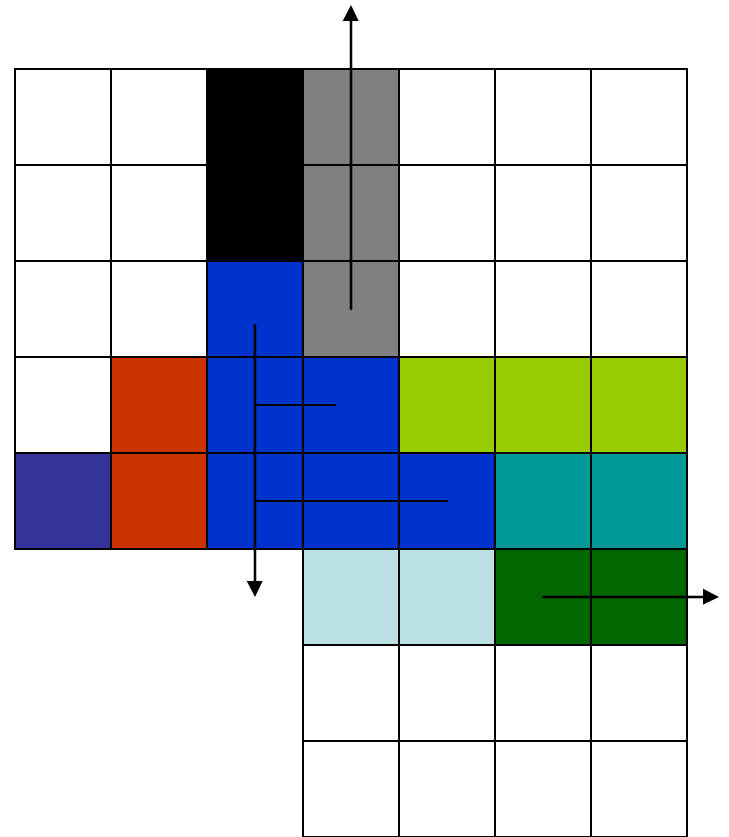
Right only

106-APX



A Key Idea

- Voronoi decomposition
 - closest boundary pixel edge
 - tie-breaking
 - clear Voronoi-cell-by-Voronoi-cell
- Lower Bounds
 - snow amount
 - distance to boundary



Lower Bounds

- snow LB

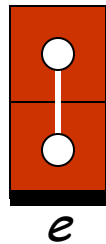
$snowLB(R) =$ # of pixels of region R with snow

- distance LB

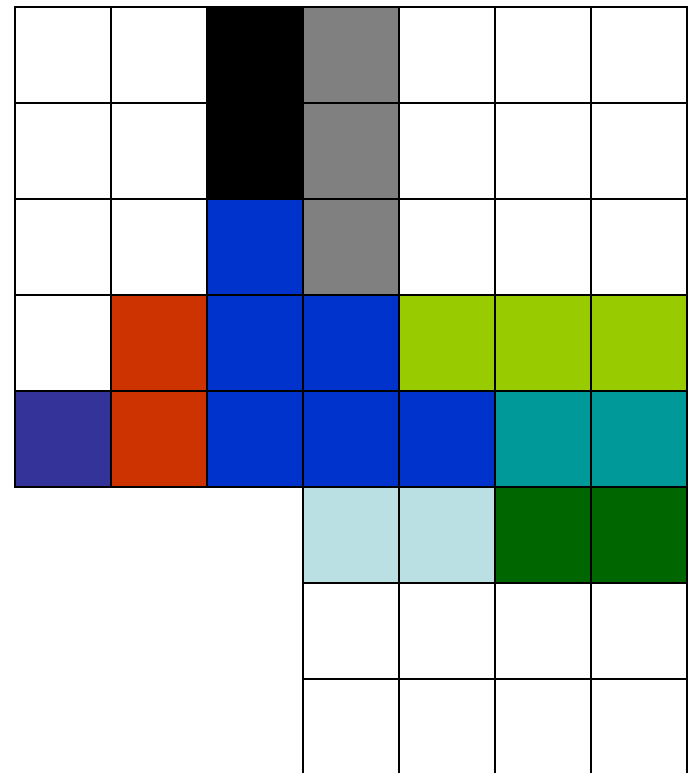
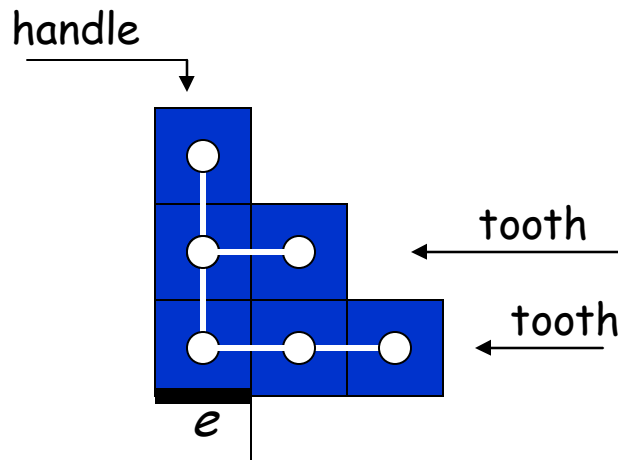
$distLB(R) = \frac{1}{D} \sum_{\text{pixel} \in R}$ [distance from pixel to the boundary]

Boundary Cells Are Of Two Types (by our tiebreaking rule)

- Lines



- Combs



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



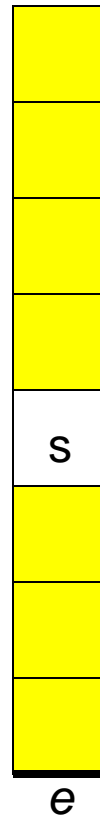
Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Line-clearing ($D=3$)

- Move up D ,
doing back
throws
- U-turn
- Forward throw
moving down to
boundary



Cost of Line L

- Each D -full pass (clearing D snow units)

$$\text{distLB}(\text{cleared}) = (h+1+\dots+h+D)/D$$

$$\sim h+D/2$$

$$\text{cost} = 2(h+D)$$

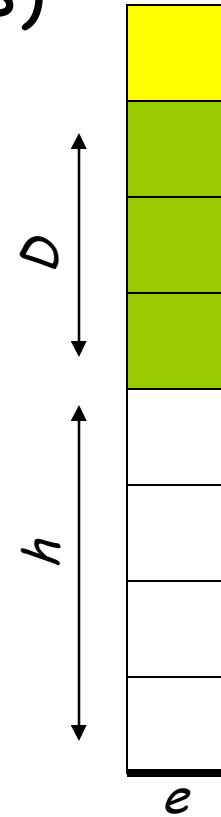
$$\leq 4 \text{distLB}(\text{cleared})$$

- The pass that is not D -full

$$\text{cost} \leq 2 \text{snowLB}(L)$$

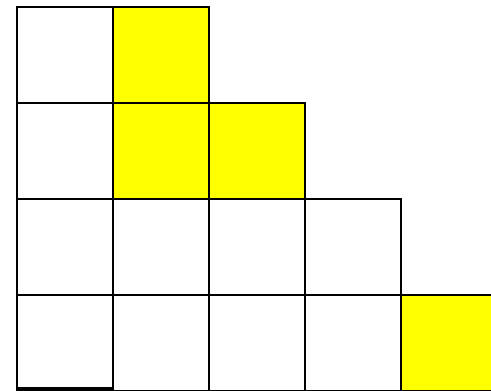
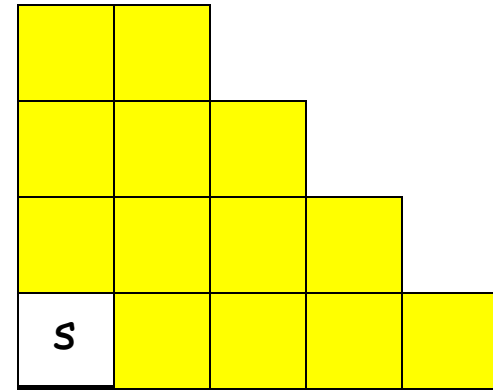
- Total cost to clear line L

$$\text{cost}(L) \leq 4 \text{distLB}(L) + 2 \text{snowLB}(L)$$



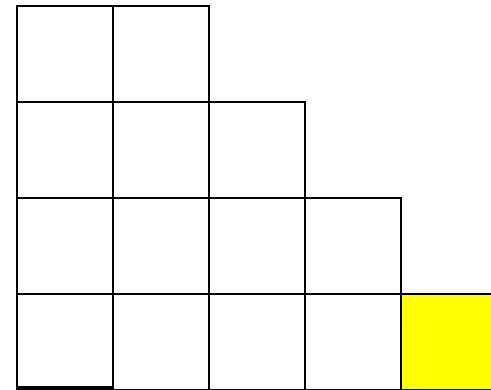
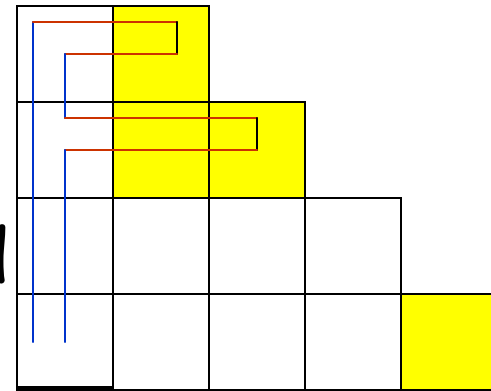
Clearing a Comb ($D = 3$)

- First clear whatever lines we can using D -full passes (where clear D units of snow)
- $cost(L) \leq 4 \text{ distLB}(L)$
- Now the comb is ready for another operation: "brush-ready".



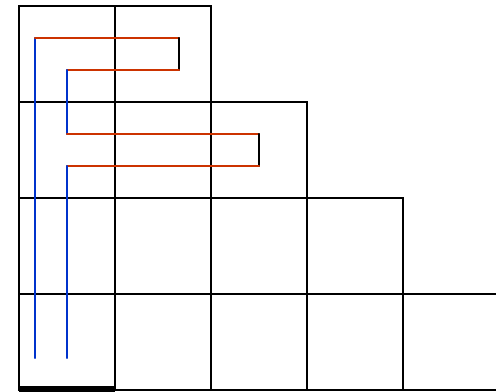
Brush Operation for Clearing Combs

- "Capacitated DFS"
 - Proceed tooth by tooth
 - until D units of snow moved
 - clear a tooth and move down



All Brushes

- $cost(brushes) =$
 $cost(blue) + cost(red)$
 $\leq 4 snowLB(comb) +$
 $2 distLB(cleared)$



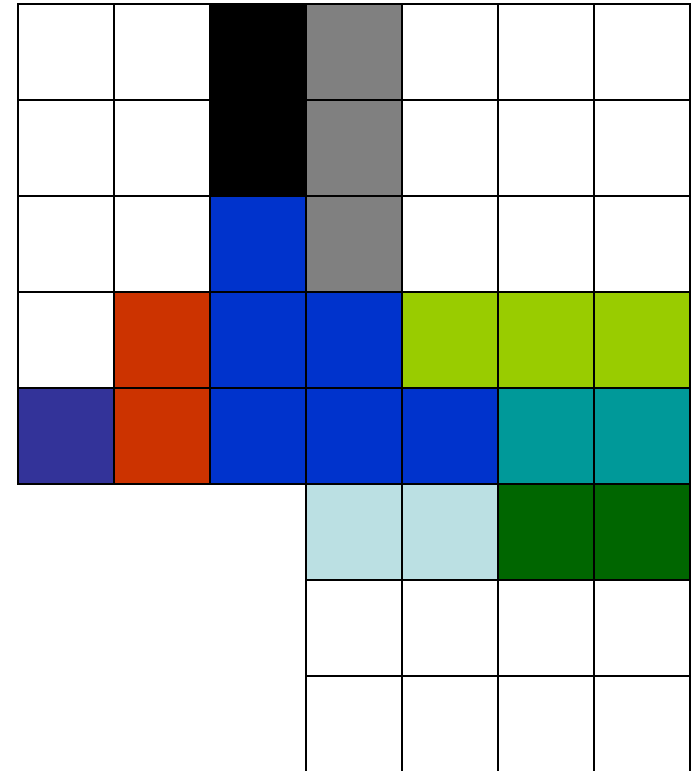
Comb Clearing

- $cost(comb) =$
 $cost(line-clearing) + cost(brushes)$
 $\leq 4 snowLB(comb) + 4 distLB(comb)$

Cost of Polygon P

Treat each vornoi cell independently...

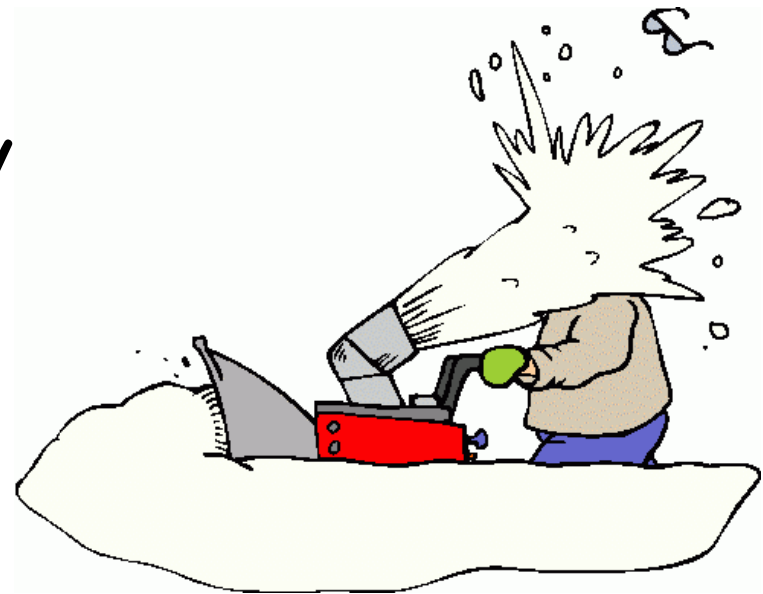
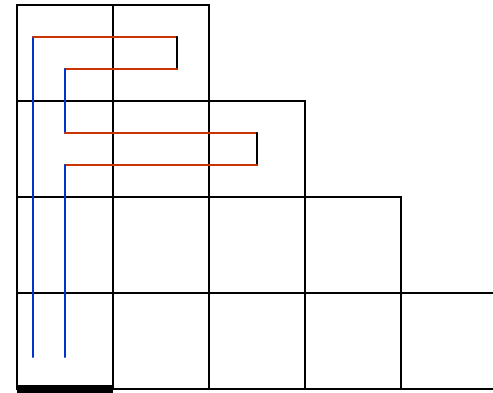
- $cost(L) \leq 4 \text{ snowLB}(L) + 2 \text{ distLB}(L)$
- $cost(comb) \leq 4 \text{ snowLB}(comb) + 4 \text{ distLB}(comb)$
- $cost(P) \leq 4 \text{ snowLB}(P) + 4 \text{ distLB}(P)$
- $OPT \geq \text{snowLB}(P), \text{distLB}(P)$



8-approx

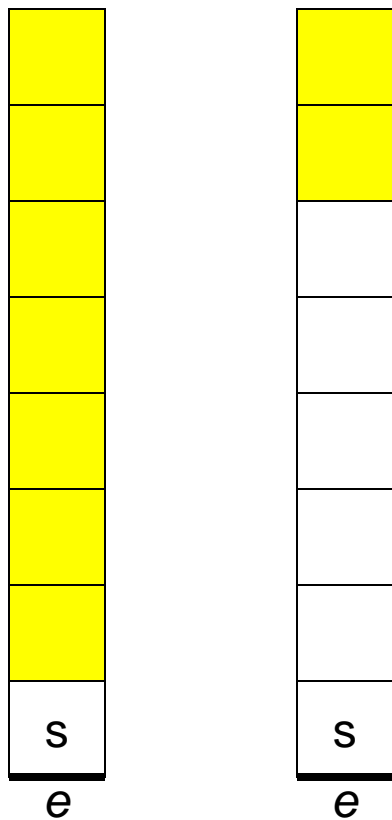
Other Throw Models

- Reduce to the default model (We found the emulations surprisingly complicated)
- Other issue: snow doesn't travel directly to its Voronoi edge

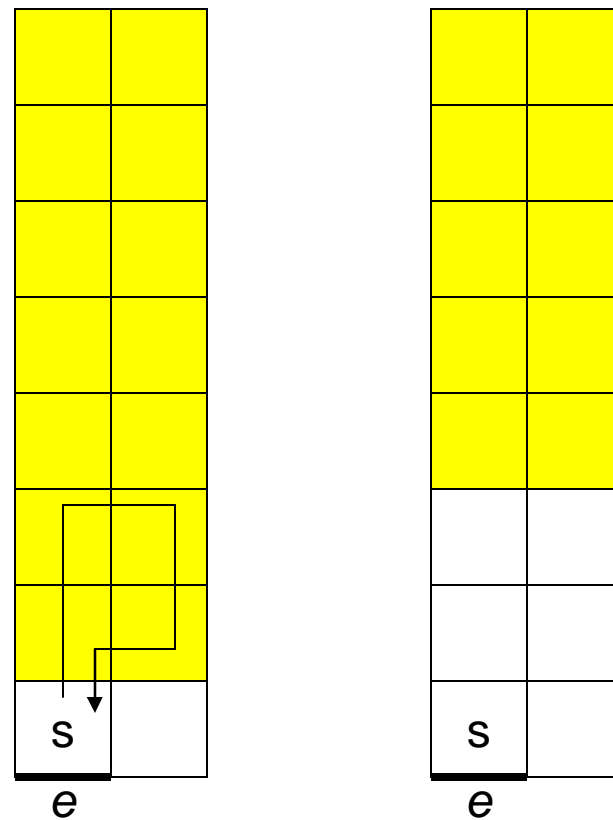


Line-clearing

- D -full pass



- $\lfloor D/2 \rfloor$ -full pass



Adjustable Throw

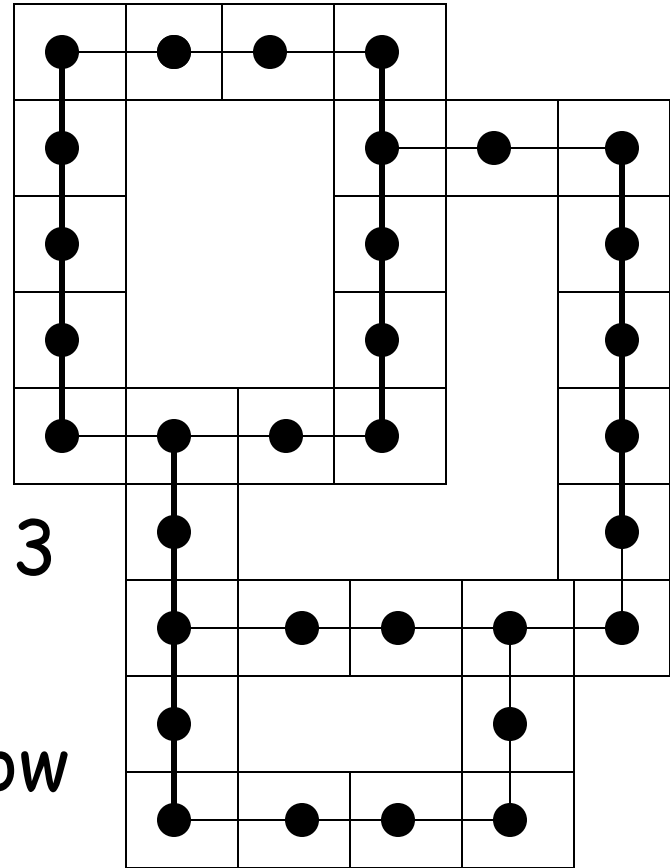
$(4 + 3D/\lfloor D/2 \rfloor)$ -*approx*

Fixed Throw

$(34 + 24D/\lfloor D/2 \rfloor)$ -*approx*
more involved emulations

Hardness

- In NP
 - by our algorithms
- NP-hard
 - Hamilton Cycle in $\text{deg} \leq 3$ grid graphs
 - default/adjustable throw
 - holes



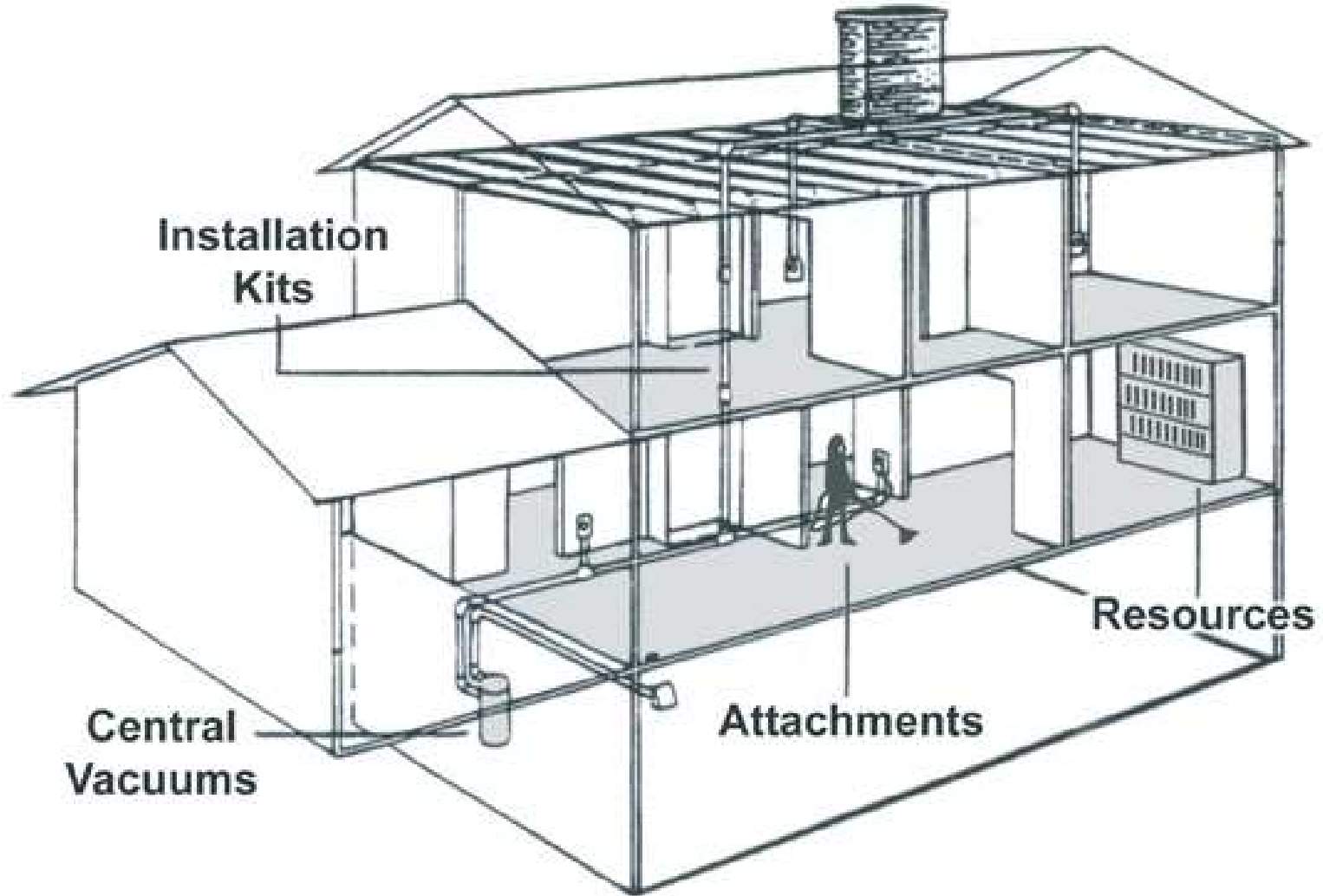
Nonuniform Initial Depth

- Straightforward generalization
 - approx factors depend on D linearly

Nonrectilinear

- Once around boundary
 - apply algorithms

Central Vacuum System



Dustpan Vac in Baseboard



"Infinite" Capacity



Vacuum Cleaner Problem

Robot



Exactly the SBP
default model

Conclusion

SBP

3 throw models

- default
 - not realistic
 - vacuum-cleaner
- adjustable
- fixed

$O(1)$ -approx

NP-complete

- default/adjustable
- holes

Open

- Hardness
 - simple polygon
 - fixed throw

- Improve approx factors

Turn Cost



Throwing >1 away



Multiple SBs



The Online Problem

