The Snow Blower Problem

The Stony Brook Algorithms Reading Group

November 2, 2006

Abstract

We introduce a new optimization problem, the "snow blower problem" (SBP), which is closely related to milling problems and some material handling problems. The objective in the SBP is to compute a good path for a snow blower to be able to remove "snow" from a region (driveway, sidewalks, etc), while taking into account the fact that when a snow blower passes over a region, it displaces the snow to a nearby adjacent region, thereby making the snow deeper in some places. We assume that the snow blower can operate effectively only for snow up to a certain depth. Then, without careful planning, use of a snow blower can result in piling snow too deeply in some areas, where it is then not feasible to use the snow blower to clear the deep snow.

We give the first algorithmic study of this problem, showing how, in polynomial time, one can decide if a network of sidewalks can be cleared, and giving polynomial-time approximation algorithms for optimally removing snow from regions, under various assumptions about the operation of the snow blower.

Our results are applicable also to some versions of a material handling problem, in which the goal is to rearrange cartons on the floor of a warehouse.

1 Introduction

On March 5-6, 2001, a large snow storm hit the northeastern United States. Prompted by his experience after the storm in using a powered snow blower to clear the driveway and sidewalks at the Arkin-Mitchell Manor on Long Island, Joe Mitchell posed the following question at a March 9 algorithms seminar:

> How does one optimally use a snow blower to clear a given polygonal region?

We study this problem, dubbed the *Snow Blower Problem* (SBP), giving a precise formulation and a number of first algorithmic results.

1.1 Related Work

Our problem is closely related to the milling and lawn mowing problems, which have been studied extensively by many authors in the NC-machining literature as well as the computational geometry literature. See Held et al [**?**]. The main distinction in the SBP problem is that the material that is being removed by "milling" or "mowing" does not simply "evaporate", fall to the ground as fine debris, or disappear; instead, it is moved to the side of the snow blower and may have to be handled again later if it falls on a region that ultimately must be cleared.

Our problem is also closely related to material handling problems in operations research, in which the goal is to rearrange a set of objects (e.g., cartons) within a storage facility, while satisfying constraints on how high one is allowed to stack the objects (so that they are not crushed or do not topple over). See [**?**].

1.2 Problem Formulation

Let P denote a connected polygonal domain having n vertices and h holes. Often, we will consider the *integral orthohedral* version of our problem in which the edges of P are parallel to the x- and y-axes, and the vertices of P have integral coordinates. In this case, P is a union of unit square *pixels* whose center points define a *grid graph*, $G(P)$, whose edges link pixel center points that are at unit distance from each other.

We let B denote the shape of the "footprint" of the snow blower; B is the region under the snow blower that gets cleared of snow by the snow blower. If the center of the snow blower is at position (x, y), we let $B(x, y)$ denote the region that is cleared. We generally assume that B is a unit square.

There may be constraints on the allowed motions of the snow blower. Often we will assume that the motion of the snow blower is rectilinear, along segments that are axis-parallel.

The throw direction: fixed or adjustable.

The maximum allowed depth, D, that can be cleared using the snow blower. Assume the initial depth is 1 everywhere within P. (More generally, we may consider varying depths of initial snow.)

The distribution of thrown snow: into a single pixel or a larger region. We must specify the *throw region*, T, which will depend on the current location and heading of the snow blower. In the discrete (pixel) model, T consists of one or more pixels in some (fixed or variable) spatial relationship with respect to the current pixel and the direction from which that pixel was entered.

Costs: there may be a different cost for motion on a cleared region than for motion while clearing snow.

Snow shovel problem: explain this variant: A shovel has a capacity s. We can use it to scoop up s units of snow and carry it elsewhere. This version really is an optimal transportation problem. (Is this known to be NP-hard (at least)? Probably.)

2 Clearing Sidewalks

Assume that P is an integral orthogonal polygon whose corresponding grid graph contains no cycle of length four; in other words, it is a "thin" (width-1) sidewalk network. (No two-by-two square fits within P.)

2.1 Adjustable Throw Direction

We consider first the case in which the throw direction is adjustable (to be forward, left, or right).

We assume that $D = 1$, so the snow blower is able to clear snow of depth 1, but not of depth 2 or more.

We have an example (based on a "cross") showing that some sidewalk networks cannot be cleared.

The problem becomes that of deciding if a degree-four node of a grid graph can be "cleared", starting with the snow blower in its current position, p. This can be solved in $O(N)$ time (where

N is the number of pixels), using simple depth-first search: Search from p; degree-four nodes are dead ends; is there a degree-four node that can be reached from two distinct directions? If yes, then it can be cleared; otherwise, there is no way to clear even one degree-four node. (Claim: The order in which we do clearing does not matter, since having additional cleared pixels cannot hurt us.) In fact, the time complexity is linear in n, the number of vertices of P (which is proportional to the number of nodes of the grid graph of degree 3 or 4).

Theorem 1 *We can determine in $O(n^2)$ time if there exists a strategy to clear a sidewalk network of complexity n. If one exists, then, within the same time bound, one such strategy can be computed.*

Remaining open questions include: Can we optimize the total motion? Can we approximate it?

9/20/01: Valentin Polishchuk observes: The arguments for Sidewalk Feasibility Problem seem applicable to driveways as well ("clearability" if max(vertex degree)<4; DFS). Is there anything peculiar to thin driveways only, i.e. a statement that would be true for sidewalks and false for driveways (may be in the Optimality Problem)? (Joe: I suspect the optimality problem behaves differently (more efficient to get better approximation for the sidewalk case, I would guess).)

2.2 Fixed Throw Direction

Assume that the snow blower removes snow and piles it to the left. For now, we assume that it piles it in the pixel immediately to the left (i.e., $k = 1$), where "left" is measured according to the incoming direction (as we enter the pixel).

We assume for now that $D = 1$.

We have an example showing that some thin polygons P (e.g., those having an appropriate "zig zag") cannot be cleared.

We want to characterize exactly when it is possible to clear a network of sidewalks.

What if $D = 2$?

2.3 Hardness

We conjecture that the optimization problem is hard.

3 Clearing Driveways

Special cases: Rectangles; simple polygons (no holes), etc.

We obtain a lower bound on the optimal amount of work (total length of the snow blower path) based on classifying each pixel according to which edge of the driveway is closest (i.e., how can the pixel of snow be moved off of the driveway in the shortest possible (in L_1 metric) way?). Summing over all pixels gives a lower bound.

Conjecture: There is a simple strategy that uses at most TWICE the lower bound.

Relationship to Chinese Postman Problem....not quite.

If we draw a map showing how OPT moves each pixel off of the driveway, then each step along the paths represents a step that the snow blower must make (entering that pixel from the appropriate direction). This is a directed edge in the grid graph, possibly with repetitions to represent the fact that we may need to enter a pixel multiple times (depending also on the capacity D). Our problem then becomes that of computing a shortest tour in a grid graph that goes through

each of a specified set of directed edges (with weights to indicate multiplicities). This is a Stacker-Crane problem (known to be hard); it has a 9/5 approximation algorithm by Frederickson et al. [**?**]. This seems to give a 9/5-approximation algorithm for our problem too. (But this still needs some proof, since we do not know a priori what the mixed graph is in the stacker-crane problem; we can define one based on the lower bound (moving each pixel of snow along a shortest path to the boundary).) Can we exploit more structure and do better?

See: Frederickson, G. N., Hecht, M. S., and Kim, C. E. (1978), "Approximation algorithms for some routing problems", SIAM J. Comp. 7, 178-193.

4 Other

Consider the "Garage Placement" problem: Where should we build the garage (and position its door) so that we can clear the driveway?

9/20/01: Valentin Polishchuk: Color all pixels (maybe even all edges of pixels to represent "entrance direction") as to weather it is possible to clear the area starting form the pixel. It can be done in $O(n^2)$ by DFS from every pixel, but may be faster? Then we would be able solve the Feasibility Problem faster (although "non-constructively"). (Joe: Why would the feasibility problem have a faster algorithm?)

5 Conclusion

References