The Discrete and Continuous Snowblower Problem

Sándor P. Fekete* Jan-Christoph Kalo* Christiane Schmidt*

Abstract

We consider the snowblower problem for a fixed throw direction. For the discrete case, introduced in [2] we improve the approxmation factor. In addition, we generalize the problem and formulate the continuous snowblower problem. The snowblower is a line of length 1 that moves overtraveled snow to the right by exactly 1 unit at any point in time. In particular, this extension allows to deal with a maximal allowed height of one ($D = 1$), as we are able to deal with reflex vertices of the polygon. For this model we present approximation algorithms for squares and polyominoes.

1 Introduction

The *snowblower problem* (SBP) was introduced by Arkin et al. [2]. Given a polygonal region it asks for a shortest tour of a snowblower such that all snow initially located in the polygonal region is moved outside. The movement of snow is modeled as a shift of a unit of "material" (i.e. snow for our imagination). At each point in time the snow may not exceed a maximum allowed height D. If you consider boxes rather than snow units, this relates to a bound on the stack height. The snowblower is modeled as a unit square that moves horizontally and vertically, only, and uses steps of length 1. Hence, integral-orthogonal polygons, polyominoes, are considered. For the outside no height bound applies. Arkin et al. [2, 3] considered several models for the direction in which the snowblower may throw the snow. In the default model snow may be thrown in any direction (left, right, forward and backward), for the adjustable-throw model the backward throw is excluded, the fixed-throw model restricts throws to the right. In this paper, we consider the fixed-throw model, only.

In the following we will refer to this problem as the *discrete snowblower problem* (DSBP), this notion is motivated by the unit steps of the snowblower and the units of snows moved in any step. This assumption makes it impossible to clear any polygon, in particular, any polyomino, with reflex vertices and $D = 1$. See Figure 1(a): the snow from the shaded pixel cannot be moved without building a stack on another pixel or the allowance to move over uncleared regions.

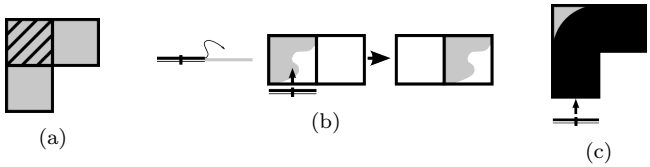


Figure 1: Snow is shown in gray. (a) The shaded pixel cannot be cleared for the DSBP and $D = 1$. (b) The snowblower in the CSBP moves overtraveled snow to the right by one unit. (c) Moving around a reflex vertex in the CSBP model: the black area is cleared if the snowblower enters the three pixels from below.

This motivated us to extend the problem: We think of the snowblower as a line of length 1 (moved around by its center point). At any point in time it will move any snow that it covers by exactly one unit to the right, see Figure 1(b). Hence, the snowblower is now able to cope with reflex vertices, see Figure 1(c), all snow is cleared over the vertex. Thus, in the new model we are able to clear polygonal regions for a maximal allowed height of 1 ($D = 1$). Again, we are interested in a shortest tour for our snowblower such that all snow in the polygonal region ends up outside of its boundary. As we use the term "short" we still need to determine the cost of a movement. Using covered distance only, we could rotate around the snowblower's center without causing any costs, though snow may be moved during this operation also. On the other hand, if we restrict to consider the area swept through, we may cover an arbitrary distance for cost zero by moving parallel to the snowblower's extension. Consequently, the cost for any section is the maximum of distance covered and area swept through. We will refer to this problem variant as the *continuous snowblower problem* (CSBP). The rest of this paper is organized as follows. Section 3 provides definitions, in Section 4 we consider the discrete snowblower problem and present approximation strategies for squares and polyominoes, we consider the same polygon classes for the continuous snowblower problem in Section 4. We conclude in the final Section 6.

2 Related Work

As mentioned in Section 1 the SBP was defined by Arkin et al. [2, 3]. The authors proved the SBP in

*Department of Computer Science, Braunschweig University of Technology, Germany, {s.fekete,c.schmidt}@tu-bs.de

the first two models to be NP-complete for polygonal domains with holes. They present approximation algorithms for all three throw models. For the default model they give a discrete Voronoi decomposition of the polyomino and present operations to clear the occurring lines and combs of this decomposition. This allows for an 8-approximation. For the other models Arkin et al. show how to emulate the operations performed in the default model by a sequence of moves. As an adjacent row needs to be used, $\lfloor \frac{D}{2} \rfloor$ pixels of snow are considered for an operation only, this results in a dependance on $D/\lfloor D/2 \rfloor$ of the approximation factors: $(4+3D/\lfloor D/2 \rfloor)$ for the adjustable-throw model and $(34 + 24D/\lfloor D/2 \rfloor)$ for the fixed-throw model.

The CSBP and even more the DSBP are closely related to milling and lawn-mowing problems, studied for example by Arkin et al. [4] and Held [5]. In particular, for the CSBP and $D = 1$ reflex vertices infer a cost higher than "normal" sides, so treating these is closely related to milling problems with turn costs as introduced by Arkin et al. [1].

3 Notation and Preliminaries

We consider a polygon P, for this paper P is either an integer square or a polyomino. In Section 4 we keep the notation from Arkin et al. [3].

4 Discrete Snowblower Problem

Polyominoes. We use the notation from [3] in this section. Hence, the two lower bounds, the number of pixels covered with snow, the *snow lower bound* $\text{snow}(R)$, and the *distance lower bound*

$$\text{dist}(R) = \frac{1}{D} \sum_{p \in snow(R)} d(p, \delta P) \qquad (1)$$

where $d(p, \delta P)$ denotes a pixel's p shortest distance to a boundary pixel plus 1, are the same. In addition, we consider the same Voronoi decomposition as presented by Arkin et al.: for a pixel $p \in P$ $\text{Vs}(p)$ denotes the element of δP closest to p (where we use the same tie-breaking rules as Arkin et al.), a Voronoi cell of a side $e \in \delta P$ is defined as $\text{Vor}(e) = \{p \in P \mid \text{Vs}(p) = e\}$. Any Voronoi cell is either a line (with a path as the dual graph), or a comb, or a double sided comb. For a comb the dual graph has a spanning tree with one vertical path through the "handle" and horizontal paths through the "teeth". Emphasizing this underlying graph structure $\text{Vor}(e)$ is also denoted by $T(p)$ for the boundary pixel p, this denotes a unique tree, that is a line or a comb. We need to clear the resulting lines ($\mathcal{L}$) and combs ($\mathcal{C}$) as well. Note that each shortest path in this decomposition bends at most twice - once to the right, once to the left. For our algorithm, we first clear all boundary pixels and reflex vertex pixels with one run along the boundary (this is possible for $D \geq 2$). For a line $\mathcal{L}$ we consider the line and the adjacent pixels, see Figure 2(a). Due to the cleaning of the boundary, $\mathcal{L}$'s base as well as the two pixels marked by a cross (if both exist, otherwise only the top pixel) are clear. Either of them is a boundary pixel, we will use it to move the snow outside of P. We start with clearing the line to the adjacent pixels, this move is denoted by the dark gray dashed line. As we have two rows of pixels—but charge to one line—we split the line in sections of length $\lfloor \frac{D}{2} \rfloor$, indicated by the horizontal lines in Figure 2(a). Then, we only need to make sure that we transport the shaded pixels—the pixels within distance $i \cdot \lfloor \frac{D}{2} \rfloor$, $i \in \mathbb{N}$ on $\mathcal{L}$—(by now located in the row to the right) to the boundary, all other pixels in the same interval will be cleared "automatically" (they are carried along). In case we consider a straight line we need three steps to push the snow one pixel further to the boundary, indicated by the bold black line in Figure 2(a). Left bends only decrease the necessary number of steps, for a right bend we need 5 steps, see Figure 2(b).

With the notation from [2] we let $\mathcal{L} \mid \mathcal{J}$ denote the line $\mathcal{L}$ with the first J pixels clear, $l = \mid \mathcal{L} \mid$, $l - J = k' \lfloor \frac{D}{2} \rfloor + r'$. $(\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor}$ denotes the first $k' \lfloor \frac{D}{2} \rfloor$ pixels of $\mathcal{L} \mid \mathcal{J}$, $\mathcal{L}_{r'}$ the last r' pixels. We spend $2(l-1)$ for the initial line shift, at most $5 \sum_{i=1}^{k' \lfloor \frac{D}{2} \rfloor} (J + i \cdot \lfloor \frac{D}{2} \rfloor)$, that is, 5 times the boundary distance of the shaded pixels except for the last (not within distance $i \cdot \lfloor \frac{D}{2} \rfloor$, $i \in \mathbb{N}$), to clear the snow up to a depth of $k' \lfloor \frac{D}{2} \rfloor$ into $\mathcal{L}$ (that is, for clearing $(\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor}$) and at most $5(l-1)$ to clear the remaining r' units of snow. We have:

$$\begin{aligned} \text{dist}((\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor}) &= \frac{1}{D} \sum_{i=1}^{k' \lfloor \frac{D}{2} \rfloor} (J + i) \\ &= \frac{\lfloor \frac{D}{2} \rfloor}{D} (k'J + \left\lfloor \frac{D}{2} \right\rfloor \frac{k'(k'+1)}{2}) \\ \text{snow}(\mathcal{L} \setminus p) &= l - 1 \end{aligned}$$

$$\begin{aligned} \text{cost}(\mathcal{L} \mid J) &= [\text{cost}((\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor})] + [\text{cost}(\mathcal{L}_{r'})] \\ &= [2(l-1) + 5 \sum_{i=1}^{k' \lfloor \frac{D}{2} \rfloor} (J + i \cdot \left\lfloor \frac{D}{2} \right\rfloor)] \\ &\quad + [5(l-1)] \\ &= 7(l-1) + 5 \cdot (k'J + \left\lfloor \frac{D}{2} \right\rfloor \cdot \frac{k'(k'+1)}{2}) \end{aligned}$$

$$\Rightarrow \text{cost}(\mathcal{L} \mid J) \leq 5 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}((\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor}) + 7 \cdot \text{snow}(\mathcal{L} \setminus p).$$

For a line where all clearing operation are $\lfloor \frac{D}{2} \rfloor$-full passes (a pass that clears $\lfloor \frac{D}{2} \rfloor$ units of snow) the $5(l-1)$ are omitted, we can bound the $2(l-1)$ by the distance

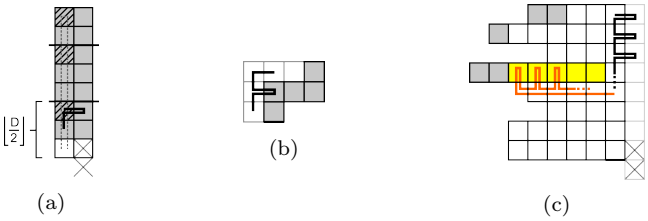


Figure 2: (a) Clearing a line $\mathcal{L}$, the left column of pixels. (b) 5 steps are necessary to move snow one pixel further to the boundary along a line with a right bend. (c) A comb $\mathcal{C}$. Snow is depicted in light gray in all Figures.

lower bound as well and we obtain:

$$\begin{aligned} \text{cost}(\mathcal{L} \mid J) &\leq 5 \sum_{i=1}^{k' \lfloor \frac{D}{2} \rfloor} (J + i \cdot \left\lfloor \frac{D}{2} \right\rfloor) + J + 2k' \left\lfloor \frac{D}{2} \right\rfloor \\ &\leq 7 \sum_{i=1}^{k' \lfloor \frac{D}{2} \rfloor} (J + i \cdot \left\lfloor \frac{D}{2} \right\rfloor) \\ \Rightarrow \text{cost}(\mathcal{L} \mid J) &\leq 7 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}((\mathcal{L} \mid J)_{\lfloor \frac{D}{2} \rfloor}) \end{aligned}$$

For a comb we clear $P \subseteq \mathcal{C}$ with $\lfloor \frac{D}{2} \rfloor$-full passes. That is, while there is a line $\mathcal{L} \subseteq \mathcal{C}$ rooted at p, such that $\text{snow}(\mathcal{C}) \geq \lfloor \frac{D}{2} \rfloor$, we execute $\lfloor \frac{D}{2} \rfloor$-full passes on $\mathcal{L}$. The remaining pixels $\mathcal{B} \subseteq \mathcal{C}$ (indicated in gray in Figure 2(c)) are cleared collecting $\lfloor \frac{D}{2} \rfloor$ units of snow, starting from the highest teeth still covered with snow. These moves are also denoted as brushes: for each brush $b = 1, \ldots, B$, t_b and $t_{b'}$ are the first and the last tooth visited during brush b. Hence, each tooth is entered at most twice (no full passes were possible anymore), at a cost of 4 charged to each pixel, see the orange pass indicated in Figure 2(c) used to clear the yellow pixels. The snow is passed on to the adjacent pixel row. Thus, we need at most $2 \cdot 4 \cdot \text{size}(\mathcal{C} \setminus \mathcal{H})$ moves for the teeth, for the handle we need at most $2 \cdot 3 \cdot \sum_{b=1}^{B} (t_b - 1) \leq 6 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}(\mathcal{B})$ moves. Thus

$$\text{cost}(\mathcal{B}) \leq 6 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}(\mathcal{B}) + 8 \cdot \text{snow}(\mathcal{C} \setminus \mathcal{H})$$

As $P \cap \mathcal{B} = \emptyset$:

$$\text{cost}(\mathcal{C}) \leq 7 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}(\mathcal{C} \setminus p) + 8 \cdot \text{snow}(\mathcal{C} \setminus p)$$

Again, as in [2], we consider the disjoint trees $T(p_i), i = 1, \ldots, m$, of the Voronoi decomposition. We have at most $7 \cdot \frac{D}{\lfloor \frac{D}{2} \rfloor} \text{dist}(T(p_i) \setminus p_i) + 8 \cdot \text{snow}(T(p_i) \setminus p_i)$ for each $T(p_i) \setminus p_i$ and $2 \cdot \text{snow}(\cup_{i=1}^{m} p_i)$ for the start. We conclude:

Theorem 1 *For a snowblower that can throw only to the right, a $(8 + \frac{7D}{\lfloor \frac{D}{2} \rfloor})$-approximation to clear a simple polyomino can be found in polynomial time.*

Squares. For a $n \times n$ square, S_n, and a snow height of $D = 1$, the distance lower bound is:

$$\begin{aligned} \text{dist}(S_n) &= \sum_{i=1}^{\frac{n}{2}} (2i)^2 = \frac{1}{6}n^3 + \frac{1}{2}n^2 + \frac{1}{3}n \\ &= \binom{n+2}{3} \end{aligned}$$

We present a strategy that starts with moving in spirals (from the layer of boundary pixels to the center of S_n). After completing one spiral, we move to the outmost layer again, this invokes cost of at most n. Moving along these spirals creates 4 triangles still covered with snow, see Figure 3(a). With each spiral those triangles become smaller. At some point, we need fewer steps for moving back and clearing the next layer of the current triangle than for moving to the counterclockwise next triangle (that is for following the spiral pattern). The boundary layers of the triangles have length $(n - 2i), (n - 2i - 1), (n - 2i - 1)$, and $(n - 2i - 2)$. The distance from the endpoint of the first triangle to the vertex pixel is i, from the vertex pixel to the second triangle $i + 1$. Walking back and to the next layer infers a cost of layer length plus 2. Hence, for $(n - 2i + 2) < i + i + 1 \Leftrightarrow i > \frac{n+1}{4}$ we switch from the spiral pattern to clearing one triangle after the other.

For the spiral movements we have a cost of at most:

$$\begin{aligned} & n^2 \cdot (\frac{n+1}{4}) - n \cdot \frac{n+1}{4} - \sum_{i=0}^{\frac{n+1}{4}-1} (2i)^2 \\ = \; & (\frac{1}{4} - \frac{1}{48})n^3 + \frac{9}{16}n^2 + \frac{13}{48}n + \frac{1}{16} \end{aligned}$$

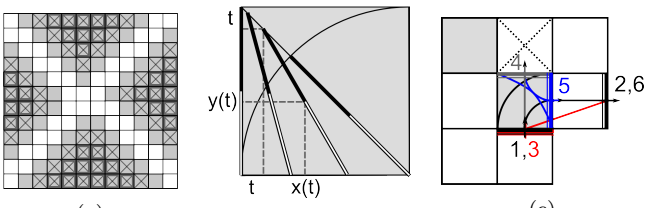
The triangle clearing costs at most (the second term bounds the cost for moving from one triangle to the next):

$$\begin{aligned} & 4 \sum_{i=1}^{\frac{n-1}{4}} 2i(\frac{n}{2} - 2i + 2) + 3(\frac{n}{2} + \frac{n+1}{4} + 1) \\ = \; & \frac{1}{24}n^3 + \frac{1}{2}n^2 + \frac{17}{24}n + \frac{5}{2} \end{aligned}$$

Altogether, the cost for our strategy, ALG, can be bound by:

$$\text{cost}(\text{ALG}) \leq \frac{13}{48}n^3 + \frac{17}{16}n^2 + \frac{155}{48}n + \frac{39}{16}$$

Theorem 2 *For the continuous snowblower problem in $n \times n$-squares with snow height $D = 1$ a solution with cost at most $\frac{13}{48}n^3 + \frac{17}{16}n^2 + \frac{155}{48}n + \frac{39}{16}$ can be found in polynomial time. For $n \geq 8$ this yields a 2-approximation. Asymptotically, we obtain a $\frac{13}{8}$-approximation.*



(a) (b) (c)

Figure 3: (a) Snow locations in a sqaure S_n after moving 1 (light gray), 2 (cross) and 3 (dark gray squares) spirals. (b) Continuous movement of the snowblower for a non-reflex vertex. One half of the snowblower is black, the other half white. (c) Clearing a reflex vertex in a polyomino: the snowblower moves from position 1 to 6, all positions are given with a direction, denoted by a small arrow.

5 Continuous Snowblower Problem

For the continuous snowblower problem each section of the snowblower's path infers a cost of the maximum of distance covered and area swept through. We consider a maximal allowed snow height of $D = 1$ for this problem. Let s_i denote the position of the center point (with a given orientation of the snowblower) and let $\mathrm{SB}(s_i)$ denote the set of all points of P covered by the snowblower with the center point at s_i. Then:

$$\text{cost(move from } s_1 \text{ to } s_2) = \max\{\text{length}(C(s_1, s_2)), A(\{p \in P \cap \mathrm{SB}(x), x \in [s_1, s_2]\})\}$$

where $C(s_1, s_2)$ denotes the curve followed while moving from s_1 to s_2.

As for the discrete case, we can give a lower bound depending on each snow particle's shortest distance to the polygon boundary, let this be denoted by $d(f)$. A snow particle needs to be transported stepwise, each time it will cover a distance of 1. Consequently, we obtain:

$$\mathrm{dist}(F) = \int_F d(f)df$$

For this paper we restrict to squares and polyominoes. Hence, we need to give a strategy to cover reflex and non-reflex vertices. We start with the latter, see Figure 3(b) (the lower right corner is the vertex of the polygon): the center point's movement indicated by the 4 positions of the snowblower is continuous ($x'(t) > 0$ for positive t). The area covered by this and the mirrored movement for the upper triangle is 1. For the distance covered by the center point, we describe the curve the center point follows by a parametric equation: $t \to (x(t), y(t)) = (t + \frac{1}{2}\sqrt{1-(1-t)^2}, \frac{1-t}{2}), \quad t \in [0, 1 - \frac{1}{\sqrt{2}}]$. The distance the center point covers along this parameterized curve is:

$$L(0, 1 - \frac{1}{\sqrt{2}}) = \int_0^{1-\frac{1}{\sqrt{2}}} \sqrt{\dot{x}^2 + \dot{y}^2}dt = \int_0^{1-\frac{1}{\sqrt{2}}} \sqrt{(1 + \frac{1-t}{2\sqrt{1-(1-y)^2}})^2 + (-\frac{1}{2})^2}dt$$

This integral can be solved numerically, and we obtain $L(0, 1 - \frac{1}{\sqrt{2}}) \approx 0.664988$. Consequently, the cost for clearing a non-reflex vertex using the movement shown in Figure 3(b) is $\max\{1, 2 \cdot 0.664988\} = 1.329976 =: \mathrm{cost}(NR)$.

Squares. We consider a $n \times n$ square, S_n. Hence, $\mathrm{dist}(S_n) = \frac{1}{6}n^3$, $\mathrm{snow}(S_n) = n^2$. Our strategy spirals from the boundary to the center, each time reducing the depth by 1. That is, we move along all pixels within distance 1 to the boundary, applying the non-reflex pixel strategy from above, then we move to the layer with pixels of distance 2 to the boundary, and so on. Using the non-reflex strategy described above we disperse the snow initially located on one pixel onto 3 pixels on the lower layer. These pixels are located along the angle bisectors. Hence, we do not obtain the structure of 4 triangles as in the discrete case. Consequently, we keep on moving in the spiral pattern. Each time a spiral is performed, all snow is moved a distance of 1 closer to the boundary, thus, we can decrease the depth of our spiral. The movement from the end of one spiral to the start of the next is not longer than n. The resulting cost is:

$$\begin{aligned} &\sum_{j=0}^{\lfloor \frac{n}{2} \rfloor} 4(n - 2j - 2 + \mathrm{cost}(NR))(\left\lceil \frac{n}{2} \right\rceil) \\ &\qquad + \mathrm{cost}(NR) + \frac{n^2}{2} \\ \leq\ &\frac{n^3}{3} + \frac{13 - \mathrm{cost}(NR)}{2}n^2 + (5\mathrm{cost}(NR) - \frac{16}{3})n \\ &\qquad - 3\mathrm{cost}(NR) \\ \leq\ &\frac{n^3}{3} + \frac{13 - \mathrm{cost}(NR)}{2} + \frac{4}{3}n \end{aligned}$$

Theorem 3 *For the continuous snowblower problem in $n \times n$-squares with snow height $D = 1$ a solution with cost at most $\frac{n^3}{3} + \frac{13-\mathrm{cost}(NR)}{2} + \frac{4}{3}n$ can be found in polynomial time. For $n \geq 14$ this yields a 3-approximation. Asymptotically, we obtain a 2-approximation.*

Polyominoes. To give a strategy for polyominoes as well, we need to be able to treat reflex vertices. See Figure 3(c) for an overview of the steps we perform: the move from position 1 to 2 (both indicated in black, black line) invokes a cost of $\pi/4+1$ (it clears a quadrant in the light gray pixel, the other pixel is cleared completely), the move from position 2 to 3 (along the red line) invokes

a cost of $\sqrt{10}/2$, the move from position 3 to 4 (indicated in gray) costs 1, we denote the length of the move from position 4 to 5 by c_r, finally, the move from position 5 to 6 invokes a cost of 1. After following these moves from position 1 to 6 the three pixels around the reflex vertex are cleared; we denote the sum of the steps by c_0. We still need to determine c_r: the center point covers a distance of $\pi/4$ along a quadrant of radius $1/2$, the area covered is $\leq 2(1-\frac{1}{2\sqrt{2}})$, consequently, $c_r \leq 2(1-\frac{1}{2\sqrt{2}})$.

Our strategy is as follows: we clear the polyomino in layers, starting with the pixels at the boundary. Whenever we covered a segment between two vertices, we move back (on the now cleared road) and move the snow outside of the polygon. Thus, these pixels occur a cost of at most 4 times their boundary distance (move forth, back, forth and allow for the steps to the other layer). The pixels that invoke the highest cost are those on a diagonal from a reflex vertex, marked in light gray in Figure 3(c). For the second light gray pixel we start with clearing the quadrant again, for clearing this from the first light gray pixel, we have a cost of c_0, again. Before we are able to clear the remaining snow from the second light gray pixel, we need to clear the pixel labeled with a cross (we do so, by shifting the snow to the first light gray pixel again and following the steps of cost c_0), the cost of this clearing operation is not charged to the light gray pixel, but to the cross pixel (and its cost is smaller). After this, we move the remaining snow from the second light gray pixel to the pixel marked with a cross, then to the first light gray pixel, and need additional c_0 to move it out of the polyomino. Altogether, we pay $c_1 := \pi/4+2+\pi/8+c_0+3+1+c_r+1+1+2+\pi/8+c_0 = 10+\pi/2+2c_0$. We can proceed like this, each time we invoke the clearing strategy for the diagonal pixel closer to the boundary twice, thus:

$$\begin{aligned} c_0 &= 3+\frac{\sqrt{10}}{2}+\frac{\pi}{4}+c_r \\ c_{i+1} &= 2\cdot c_i+8+\frac{\pi}{2}+2(i+1) \\ \Rightarrow c_i &= 2\cdot i\cdot c_0+8+\frac{\pi}{2}+2i \end{aligned}$$

Altogether, the cost for clearing a pixel within distance i to the boundary is less or equal to $(2\cdot c_0+8+\frac{\pi}{2}+2)i = (16+\sqrt{10}+\pi+2c_r)i$. Let $v_i = \sum_{p\in P:d(p,\delta P)=i} i$. As we use an average of $i-1/2$ for pixels within distance i in the continuous distance lower bound, we have:

$$\int_F l(f)df \geq \frac{1}{2}\sum_{i=1}^{k} v_i$$

Consequently, we obtain the following bound for the cost of our algorithm, ALG:

$$\begin{aligned} \text{cost(ALG)} &\leq \sum_{i=1}^{k}(16+\sqrt{10}+\pi+2c_r)\cdot v_i \\ &\leq 2(16+\sqrt{10}+\pi+2c_r)\cdot \text{dist}(P) \end{aligned}$$

We conclude:

Theorem 4 *For the continuous snowblower problem in polyominoes with snow height $D=1$ a $(2(16+\sqrt{10}+\pi+2c_r))$-approximation can be found in polynomial time, $c_r \leq 2(1-\frac{1}{2\sqrt{2}})$.*

6 Conclusion

We considered the snowblower problem for a fixed throw direction. For the discrete case, introduced in [2] we improved the approxmation factor. In addition, we generalized the problem and formulated the continuous snowblower problem.

References

[1] E. M. Arkin, M. A. Bender, E. D. Demaine, S. P. Fekete, J. S. B. Mitchell, and S. Sethia. Optimal covering tours with turn costs. *SIAM J. Comput.*, 35:531–566, 2005.

[2] E. M. Arkin, M. A. Bender, J. S. B. Mitchell, and V. Polishchuk. The snowblower problem. In *WAFR*, pages 219–234, 2006.

[3] E. M. Arkin, M. A. Bender, J. S. B. Mitchell, and V. Polishchuk. The snowblower problem. *Comput. Geom.*, 44(8):370–384, 2011.

[4] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell. Approximation algorithms for lawn mowing and milling. *Computational Geometry: Theory and Applications*, 17(1-2):25–50, 2000.

[5] M. Held. *On the Computational Geometry of Pocket Machining*, volume 500 of *Lecture Notes in Computer Science*. Springer, 1991.