Exploring Simple Triangular and Hexagonal Grid Polygons Online

Daniel Herrmann[1] **Tom Kamphans**[2]
Elmar Langetepe[1]

[1]University of Bonn, Computer Science I, Bonn, Germany.

[2]Braunschweig University of Technology, Computer Science, Algorithms Group, Braunschweig, Germany.

19.3.2008

2008-03-25

Exploring Simple Grid Polygons

Exploring Simple Triangular and Hexagonal Grid Polygons Online

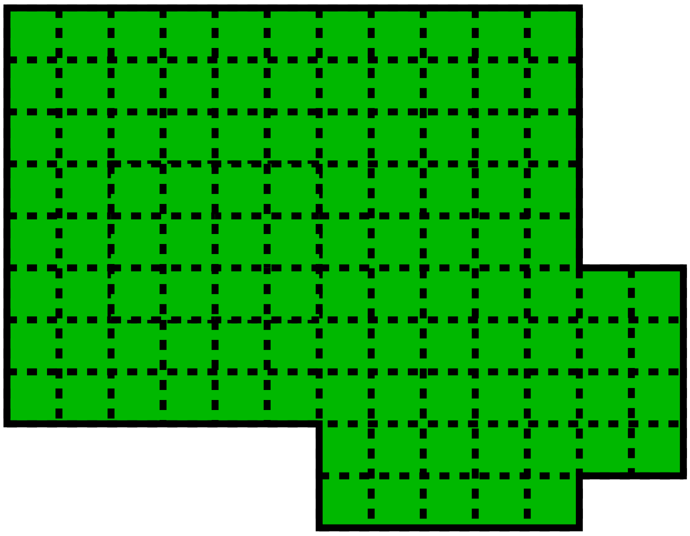
Daniel Herrmann[1] **Tom Kamphans**[2]
Elmar Langetepe[1]

[1]University of Bonn, Computer Science I, Bonn, Germany.

[2]Braunschweig University of Technology, Computer Science, Algorithms Group, Braunschweig, Germany.

19.3.2008

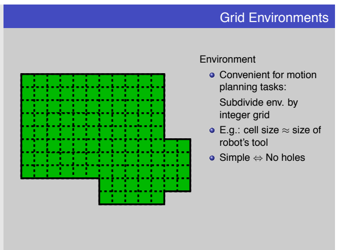
x



Environment

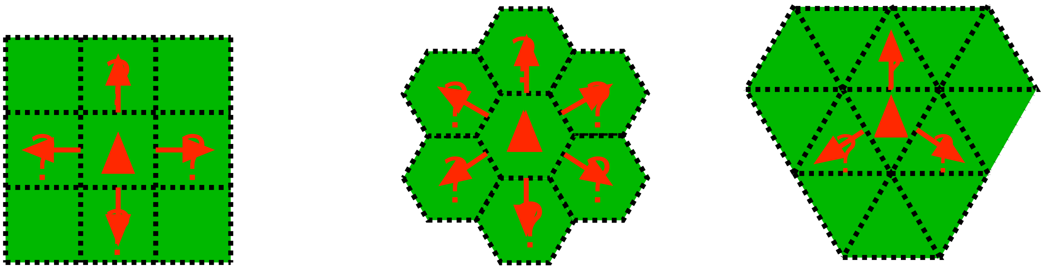
- Convenient for motion planning tasks:

  Subdivide env. by integer grid
- E.g.: cell size $\approx$ size of robot's tool
- Simple $\Leftrightarrow$ No holes

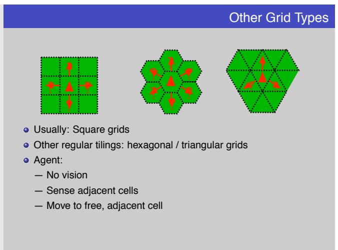


For some tasks in robot motion planning, it is convenient to subdivide the given environment by an integer grid; for example, if the agent has only limited vision or has to visit every part of the environment. Imagine, we want to mow a grass like this, then we can subdivide the environment according to the size of the tool.
We call a grid environment *simple* polygons if there are no obstacles inside the polygon.

Other Grid Types



- Usually: Square grids
- Other regular tilings: hexagonal / triangular grids
- Agent:
  - — No vision
  - — Sense adjacent cells
  - — Move to free, adjacent cell



Usually, square-shaped cells are used for such a subdivision. A natural extension is to ask for other regular tilings. There are only two other regular tilings of the plane: triangular and hexagonal grids.

We assume that the agent has no vision, but it can sense the cells that are adjacent to its current position, and that the agent can move from one cell to an adjacent cell that is part of the polygon.

The Task

Online exploration (or covering):

- Given: an unknown grid environment, P
  start cell, s, along the boundary
- Task: Find a tour that
  - visits every cell of P at least once
  - returns to the start point
  - can be computed online
  - is as short as possible
- For example: lawn mowing, cleaning

The Task
Online exploration (or covering):
Given: an unknown grid environment, P
start cell, s, along the boundary
Task: Find a tour that
visits every cell of P at least once
returns to the start point
can be computed online
is as short as possible
For example: lawn mowing, cleaning

The agent's task is to explore the whole environment. More precisely, given an unknown grid environment and a start point, we want to find a tour that visits every cell at least once and returns to start point. The environment is unknown to the robot, so we want to compute our path online. We use the length of the robot's path as quality measure, so we want to keep the path as short as possible.

Previous Work (square grids)

Offline exploration (environment is known in advance)

- With holes: NP-hard [Itai, Papadimitriou, Szwarcfiter; 1982]
- Approx. [Ntafos; 1992] [Arkin, Fekete, Mitchell; 2000]

Online exploration

- With holes:
  [Icking, Kamphans, Klein, Langetepe; 2000]: 2-competitive
  [Gabriely, Rimon; 2000]
- Without holes:
  [Icking, Kamphans, Klein, Langetepe; 2005]: $\frac{4}{3}$-competitive

Previous Work (square grids)
Offline exploration (environment is known in advance)
- With holes: NP-hard [Itai, Papadimitriou, Szwarcfiter; 1982]
- Approx. [Ntafos; 1992] [Arkin, Fekete, Mitchell; 2000]
Online exploration
- With holes:
  [Icking, Kamphans, Klein, Langetepe; 2000]: 2-competitive
  [Gabriely, Rimon; 2000]
- Without holes:
  [Icking, Kamphans, Klein, Langetepe; 2005]: $\frac{4}{3}$-competitive

There are some previous results for square grids: It is known, that the offline case where the environment is known to the robot is NP-hard for polygons with holes and there are are some approximations.
Further, there are some online strategies, both for environments with holes and environments without holes.

A Lower Bound

Problem:

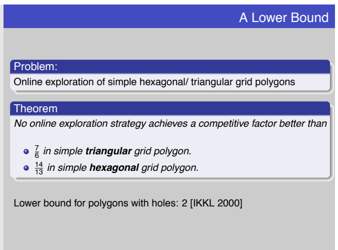
Online exploration of simple hexagonal/ triangular grid polygons

Theorem

No online exploration strategy achieves a competitive factor better than

- $\frac{7}{6}$ *in simple* ***triangular*** *grid polygon.*
- $\frac{14}{13}$ *in simple* ***hexagonal*** *grid polygon.*

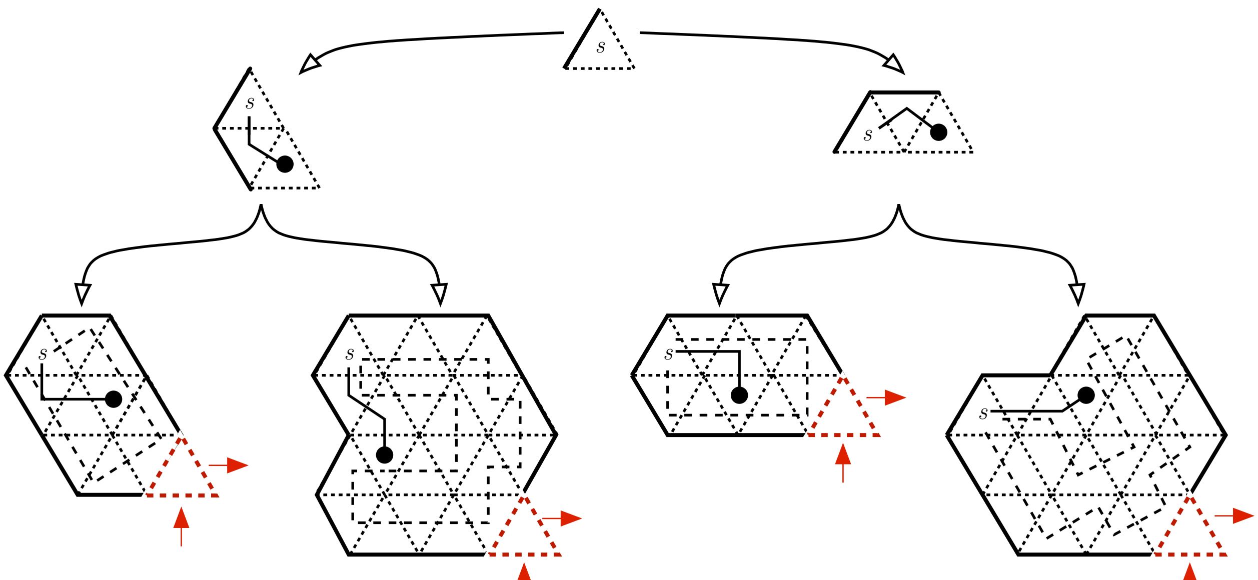
Lower bound for polygons with holes: 2 [IKKL 2000]



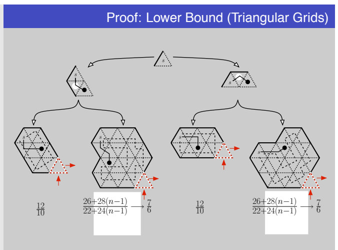
So, in the following we consider the online exploration problem for simple grid polygons on triangular and hexagonal grids..

Our first result is a lower bound on the competitive factor for our problem: We show that no online exploration strategy can achieve a factor better than 7/6 in triangular polygons and 14/13 in hexagonal grid polygons.

Proof: Lower Bound (Triangular Grids)



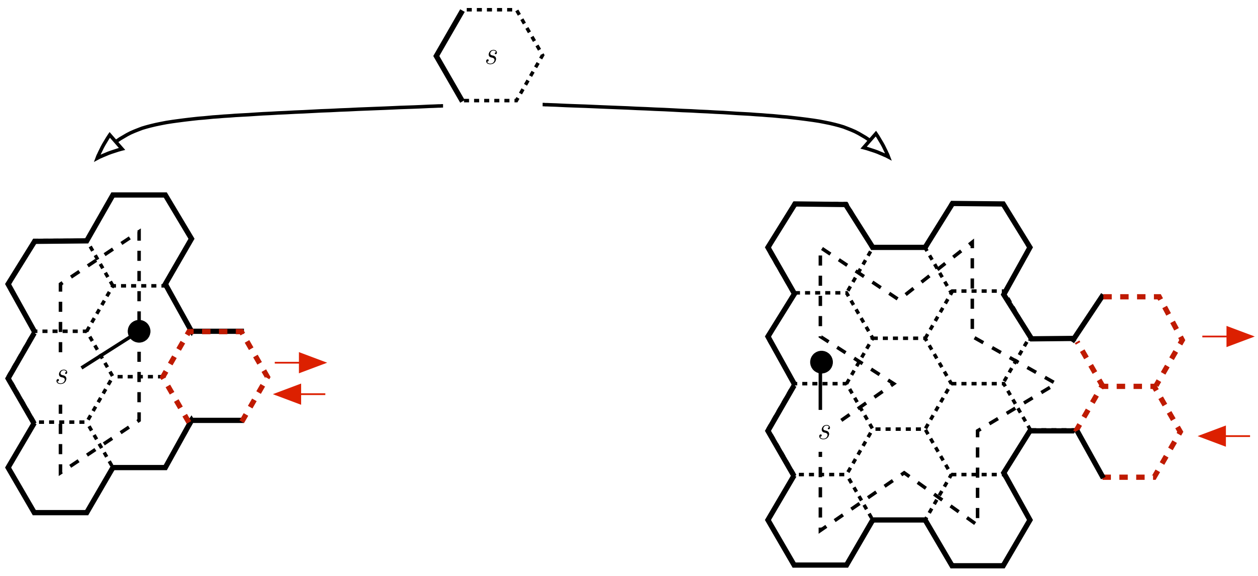
$\frac{12}{10}$ $\qquad$ $\frac{26+28(n-1)}{22+24(n-1)} \longrightarrow \frac{7}{6}$ $\qquad$ $\frac{12}{10}$ $\qquad$ $\frac{26+28(n-1)}{22+24(n-1)} \longrightarrow \frac{7}{6}$



To show the lower bound for triangular grids, let the agent start in a cell like this: it has the choice to move to the east or to the south.
Whatever the robot does, we force the robot to move east in the next step.
For the third step, the robot has two possibilities: it may walk to the south, or it may walk east. In the first case, we close the polygon like this. Now, whatever the robot does, it needs at least 12 steps, while the optimal solution needs only 10 steps. In the second case, we close the polygon like this, and get a l.b. of $\frac{26}{24}$.
On this side, we use the same blocks, just turned by 60 degree; thus, we get the same lower bounds.

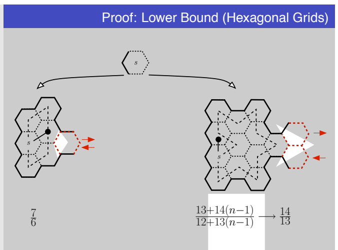
These blocks have limited size. To get a general lower bound, we have to construct polygons of arbitrary size. We can do this by repeating this construction: As soon as the robot leaves one block, it enters the start cell of the next block and the 'game' starts again. Unfortunately, both the online and the offline strategy need two more steps for the transition between the blocks, so we get a lower bound that goes to $\frac{7}{6}$.

Proof: Lower Bound (Hexagonal Grids)



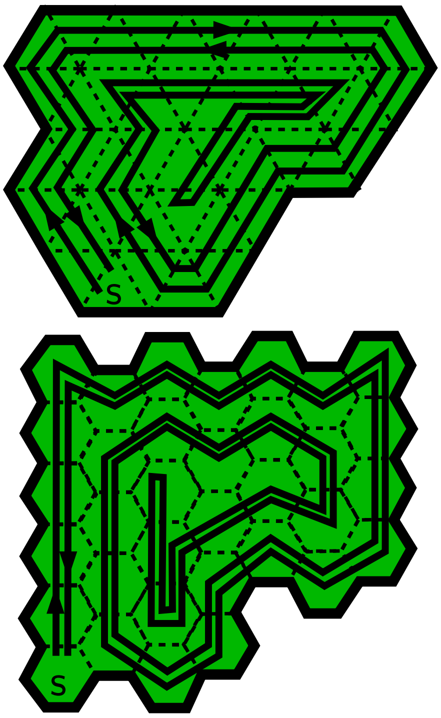
$$\frac{7}{6}$$

$$\frac{13+14(n-1)}{12+13(n-1)} \longrightarrow \frac{14}{13}$$

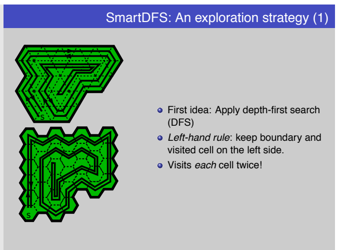


The construction for hexagonal grids is much simpler: we let the robot start in cell like this. Now, it can either leave the boundary or follow the boundary. In the first case, we close the block like this and get a lower bound of $\frac{7}{6}$. In the other case, we use this block and get $\frac{13}{12}$.

Again, we generate polygons of arbitrary size by repeating the construction using these transition cells. Altogether, we get a lower bound that goes to $\frac{14}{13}$.

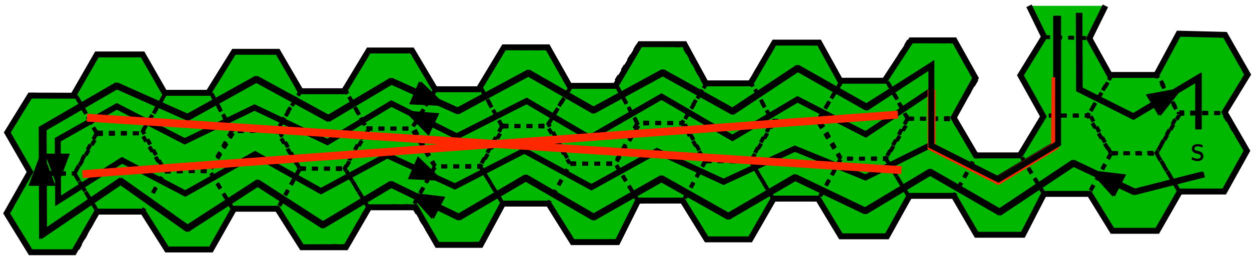


- First idea: Apply depth-first search (DFS)
- *Left-hand rule*: keep boundary and visited cell on the left side.
- Visits *each* cell twice!



Now, how can we explore such a polygon?

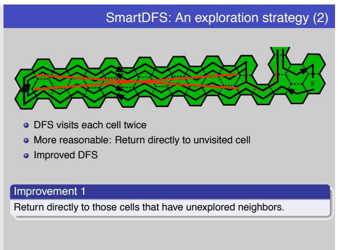
A first idea for the exploration is, to use a simple depth-first search. We use the *left-hand rule*; that is, we always keep the polygon's boundary and the visited cells on the left side of the robot. Of course, DFS visits each cell twice.



- DFS visits each cell twice
- More reasonable: Return directly to unvisited cell
- Improved DFS

Improvement 1

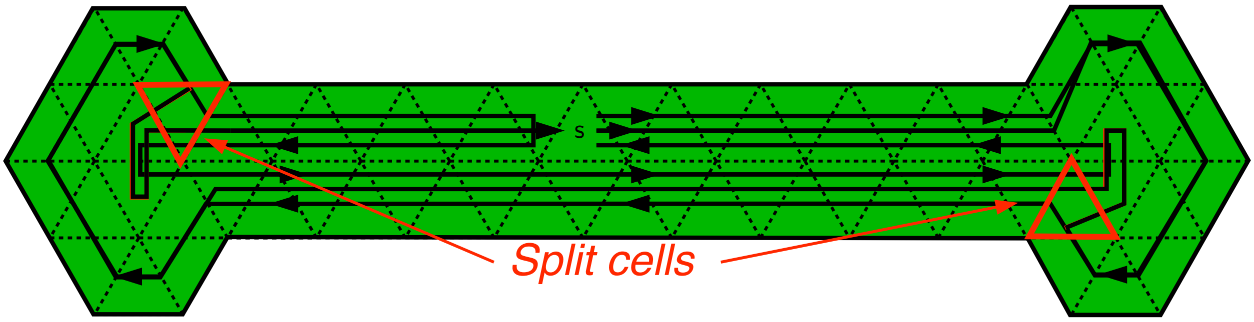
Return directly to those cells that have unexplored neighbors.



Visiting each cell twice is not very efficient, we can do better. In this example, DFS visits each cell in the long corridor twice. A more clever strategy is to omit the second visit of the long corridor and walk directly to the unexplored cell $\otimes$ so we get a path like this.

So the first improvement to DFS is to return directly to those cells that have unexplored neighbors.

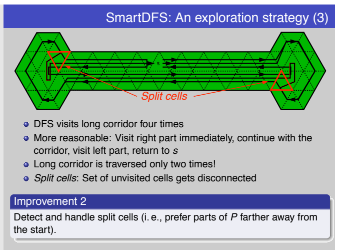
SmartDFS: An exploration strategy (3)



- DFS visits long corridor four times
- More reasonable: Visit right part immediately, continue with the corridor, visit left part, return to s
- Long corridor is traversed only two times!
- *Split cells*: Set of unvisited cells gets disconnected

Improvement 2

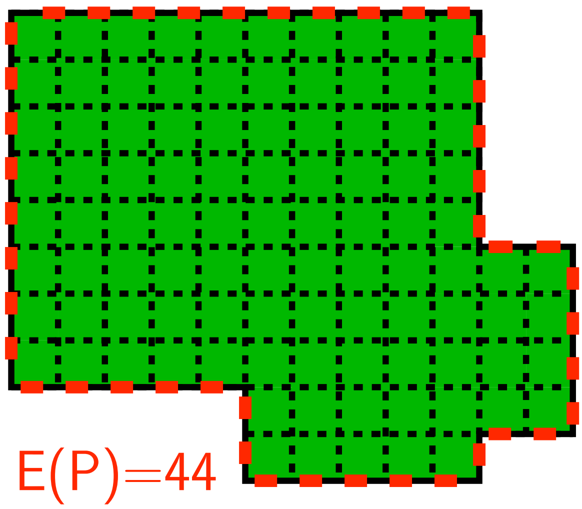
Detect and handle split cells (i. e., prefer parts of P farther away from the start).



But we still can do better. In this case, even the improved version of DFS visits the long corridor in the middle four times.
An even more smarter strategy explores the polygon up to this cell. Now, we do not follow the left-hand rule, but we explore the right part first. Then, we continue with the corridor, visit the left part and return to the start. Now, we visit the corridor in the middle only two times!
The cells on which we diverge from the left-hand rule have the property, that the unvisited cells split in two components after the cell is visited; we call cells like this *split cells*.

Thus, the second improvement to DFS is to detect split and handle split cells. Basically, we want to deal with components that are farther away from the start first.

Layer and Offset

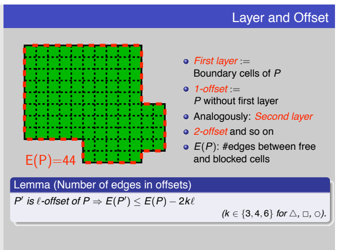


- *First layer* :=
  Boundary cells of P
- *1-offset* :=
  P without first layer
- Analogously: *Second layer*
- *2-offset* and so on
- $E(P)$: #edges between free and blocked cells

Lemma (Number of edges in offsets)

P' is ℓ-offset of $P \Rightarrow E(P') \leq E(P) - 2k\ell$

($k \in \{3, 4, 6\}$ for $\triangle$, $\square$, $\hexagon$).



We need some definitions and lemmas for the analysis of our strategy. First, we call the boundary cells of P the *first layer* of P. P without its first layer is called the *1-offset* of P. The boundary cells of the 1-offset are called the second layer of P, P without its first and second layer is called the 2-offset and so on.
We define E to be the number of edges between a free cell and a blocked cell. In this polygon, for example, we count the edges shown in red.

An important fact is that we have an upper bound on the number of edges in an offset that is substantially smaller than the number of edges in P.

Lemma (Shortest Path)

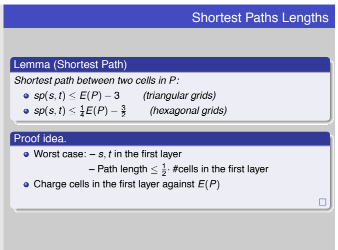
Shortest path between two cells in P:

- $sp(s,t) \leq E(P) - 3$ *(triangular grids)*
- $sp(s,t) \leq \frac{1}{4}E(P) - \frac{3}{2}$ *(hexagonal grids)*

Proof idea.

- Worst case: – s, t in the first layer
  – Path length $\leq \frac{1}{2} \cdot$ #cells in the first layer
- Charge cells in the first layer against $E(P)$

$\square$


Shortest Paths Lengths
Lemma (Shortest Path)
Shortest path between two cells in P:
$sp(s,t) \leq E(P) - 3$ (triangular grids)
$sp(s,t) \leq \frac{1}{4}E(P) - \frac{3}{2}$ (hexagonal grids)
Proof idea.
Worst case: – s, t in the first layer
– Path length $\leq \frac{1}{2} \cdot$ #cells in the first layer
Charge cells in the first layer against $E(P)$

Another important lemma gives us an upper bound on the length of a shortest path inside a grid polygon, in terms of the number of edges. The idea to show this is to assume the worst case; that is, both cells are located in the first layer of P and have maximal distance, that is half the number of cells in the first layer. Now, we just have to charge the boundary cells against the number of edges.

Upper for the Number of Steps

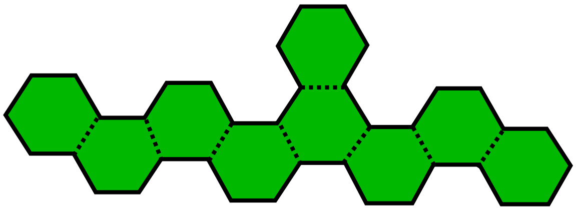
Theorem (Number of Steps)

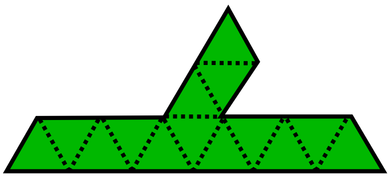
$$S(P) \leq C(P) + E(P) - 4 \qquad \textit{(Triangular grids)}$$

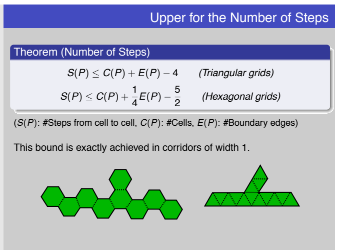
$$S(P) \leq C(P) + \frac{1}{4}E(P) - \frac{5}{2} \qquad \textit{(Hexagonal grids)}$$

($S(P)$: #Steps from cell to cell, $C(P)$: #Cells, $E(P)$: #Boundary edges)

This bound is exactly achieved in corridors of width 1.







Now, we are able to give a first performance result for our exploration strategy: The number of steps from cell to cell is bound by THIS inequation, which depends on the number of cells and the number of edges. And this bound is tight, because it is exactly achieved in polygons of width 1.

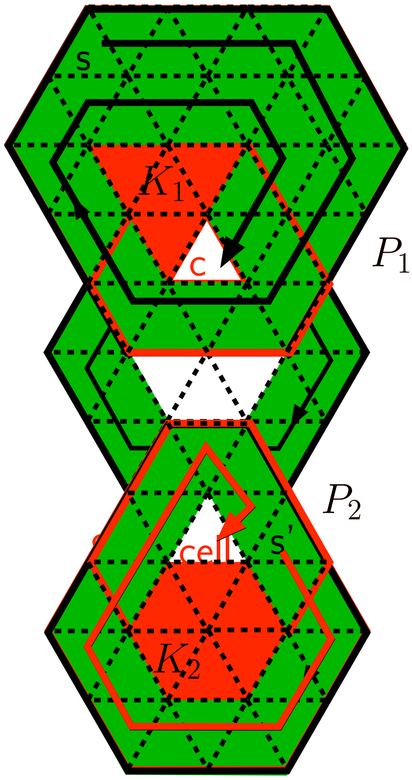
- $S(P) = \underbrace{C(P)}_{\text{Cells, i.e., necessary steps}} + \underbrace{ex(P)}_{\text{additional cell visits}}$
- Show: $ex(P) \leq E(P) - 4$ (triang.)
- Induction on the number of split cells
- Induction base: No split cell
- Visit every cell in $C(P) - 1$ steps
- Return to s in $\leq E(P) - 3$ steps (Shortest-path lemma)

Proof sketch for triangular grids
- $S(P) = \underbrace{C(P)}_{\text{Cells, i.e., necessary steps}} + \underbrace{ex(P)}_{\text{additional cell visits}}$
- Show: $ex(P) \leq E(P) - 4$ (triang.)
- Induction on the number of split cells
- Induction base: No split cell
- Visit every cell in $C(P) - 1$ steps
- Return to s in $\leq E(P) - 3$ steps (Shortest-path lemma)

The idea to show this theorem is to split the number of steps into the number of cells (the necessary steps) and the additional cell visits. Then, we show an upper bound on the number of additional cell visits by an induction on the number of split cells.

In the induction base we have no split cell, so we need $C - 1$ steps to explore the whole polygon. The length of the path back to the start is bound by our shortest-path lemma; thus, this equation holds.

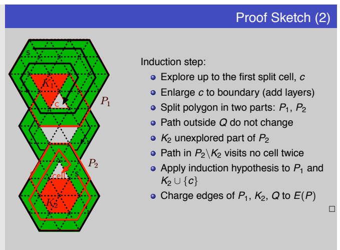
Proof Sketch (2)



Induction step:

- Explore up to the first split cell, c
- Enlarge c to boundary (add layers)
- Split polygon in two parts: P_1, P_2
- Path outside Q do not change
- K_2 unexplored part of P_2
- Path in $P_2 \setminus K_2$ visits no cell twice
- Apply induction hypothesis to P_1 and $K_2 \cup \{c\}$
- Charge edges of P_1, K_2, Q to $E(P)$

$\square$



For the induction step we proceed as follows: We explore our polygon up to the first split cell. Then we basically add some layers to c until we meet the boundary, defining a *split polygon*, Q. Then, we split our polygon in two parts **doubling some of the cells in** Q. The choice of Q ensures that the path outside Q do not change and that K_1 and K_2 are the unexplored parts of P_1 and P_2, respectively. Because c is the first split cell, the path in P_2 without K_2 visits no cell twice. And so we can simply apply the induction hypothesis to P_1 and $K_2 \cup \{c\}$ and get our result.

Competitivity, hexagonal and triangular grids

Theorem (Competitivity)

SmartDFS is $\frac{4}{3}$ competitive (i. e., $S_{\text{SmartDFS}} \leq \frac{4}{3}\, S_{\text{Optimal}}$)

Definition

Narrow passage: Corridors of width 1 or 2.

Definition

Uncritical polygon: neither narrow passages nor split cells in the first layer.

Competitivity, hexagonal and triangular grids
Theorem (Competitivity)
SmartDFS is $\frac{4}{3}$ competitive (i. e., $S_{\text{SmartDFS}} \leq \frac{4}{3}\, S_{\text{Optimal}}$)
Definition
Narrow passage: Corridors of width 1 or 2.
Definition
Uncritical polygon: neither narrow passages nor split cells in the first layer.

Using the upper bound on the number of steps, we can show a second performance result: our exploration strategy is competitive with a factor of $\frac{4}{3}$ in both types of grids. This means that the path generated by our strategy is never longer than $\frac{4}{3}$ times the optimal solution.

In the following, corridors of width 1 or 2 play an important rule, so we refer to them as *narrow passages*. More precisely, a cell belongs to a narrow passage, if we can remove this cell without changing the layer number of any other cell.

We call polygons, that have neither narrow passages nor split in the first layer *uncritical* polygons.

Competitivity in triangular grids

Lemma (Edges in uncritical triangular polygons)

For uncritical grid polygons: $E(P) \leq \frac{1}{3}C(P) + \frac{14}{3}$

Proof idea.

- Successively remove straight line of at least 3 cells (thus at most 1 edge), maintaining the property 'uncritical'
- Ends with 'diamond' polygon, $E(P) = \frac{1}{3}C(P) + \frac{14}{3}$
- Adding removed lines maintains $E(P) \leq \frac{1}{3}C(P) + \frac{14}{3}$ in every step

□

Competitivity in triangular grids
Lemma (Edges in uncritical triangular polygons)
For uncritical grid polygons: $E(P) \leq \frac{1}{3}C(P) + \frac{14}{3}$
Proof idea.
Successively remove straight line of at least 3 cells (thus at most 1 edge), maintaining the property 'uncritical'
Ends with 'diamond' polygon, $E(P) = \frac{1}{3}C(P) + \frac{14}{3}$
Adding removed lines maintains $E(P) \leq \frac{1}{3}C(P) + \frac{14}{3}$ in every step

I will sketch the proof for this for triangular polygons. For the special class of uncritical polygons, we can proof two lemmas. The first is an upper bound on number of edges in such a polygon with respect to the number of cells.

The proof idea is to successively remove a straight line of at least three cells, keeping the property that the polygon is uncritical. This decomposition ends with a diamond shaped block of cells that fulfills the equation. Now, if we reverse our decomposition process, we add at most 1 edge and at least 3 cells in every step; thus, this inequation is fulfilled in every step.

Competitivity (3)

Lemma (Exploration of uncritical polygons)

For uncritical triangular grid polygons: $S(P) \leq C(P) + E(P) - 6$.

Proof sketch

- Shown: $S(P) \leq C(P) + E(P) - 4$
- Used shortest path lemma: $sp(c, s) \leq E(P) - 4$
- Proof assumed c, s in the first layer!
- Now: c in the 1-offset
- 2 steps gained!

Competitivity (3)
Lemma (Exploration of uncritical polygons)
For uncritical triangular grid polygons: $S(P) \leq C(P) + E(P) - 6$.
Proof sketch
- Shown: $S(P) \leq C(P) + E(P) - 4$
- Used shortest path lemma: $sp(c, s) \leq E(P) - 4$
- Proof assumed c, s in the first layer!
- Now: c in the 1-offset
- 2 steps gained!

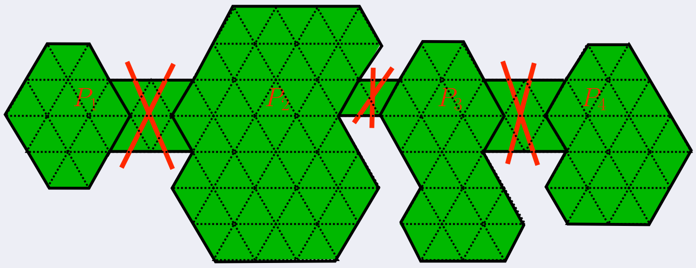
Second, we can show that we can explore uncritical polygons better than general simple polygons.
We already showed this upper bound for arbitrary simple polygons. In the induction base, we used the shortest path lemma, but to proof this lemma, we assumed that both cells are in the first layer of P.

Now, we return from a cell in the 1-offset of P to s, and so we gain two steps.

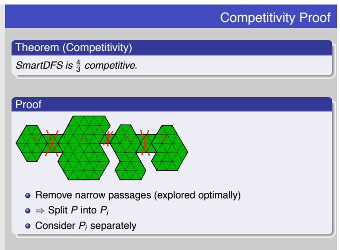
Theorem (Competitivity)

SmartDFS is $\frac{4}{3}$ competitive.

Proof



- Remove narrow passages (explored optimally)
- $\Rightarrow$ Split P into P_i
- Consider P_i separately



With these two lemmas, we can show our theorem. First, we remove all narrow passages, because they are explored optimally.
Thus, we split P into a sequence of polygons P_i, which can be considered separately without changing the competitive factor.

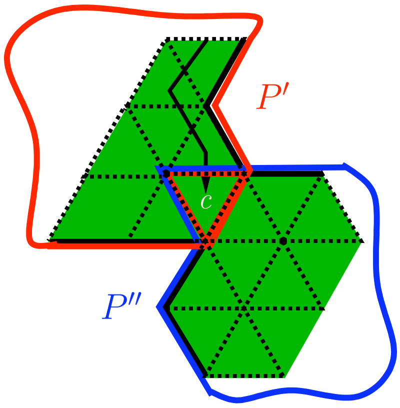
Competitivity Proof: Induction

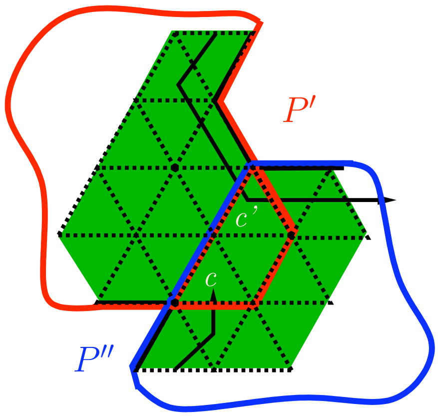
- Show $S(P_i) \leq \frac{4}{3}C(P_i) - \frac{4}{3}$
  by induction on the number of split cells in the first layer
- Ind. base: No split cell $\Rightarrow$ uncritical polygon $\Rightarrow$

$$\begin{aligned} S(P_i) &\leq C(P_i) + E(P_i) - 6 && \text{by exploration lemma} \\ &\leq C(P_i) + \frac{1}{3}C(P_i) + \frac{14}{3} - 6 && \text{by edges lemma} \\ &= \frac{4}{3}C(P_i) - \frac{4}{3} \end{aligned}$$

Competitivity Proof: Induction
Show $S(P_i) \leq \frac{4}{3}C(P_i) - \frac{4}{3}$ by induction on the number of split cells in the first layer
Ind. base: No split cell $\Rightarrow$ uncritical polygon $\Rightarrow$
$S(P_i) \leq C(P_i) + E(P_i) - 6$ by exploration lemma
$\leq C(P_i) + \frac{1}{3}C(P_i) + \frac{14}{3} - 6$ by edges lemma
$= \frac{4}{3}C(P_i) - \frac{4}{3}$

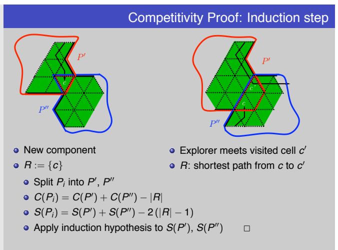
Now, we show this inequation by an induction on the number of split cells in the first layer of P_i.
If there is no split cell, then our polygon is uncritical and our result follows from the two lemmas on uncritical polygons.





- New component
- $R := \{c\}$
  - Split P_i into P', P''
  - $C(P_i) = C(P') + C(P'') - |R|$
  - $S(P_i) = S(P') + S(P'') - 2(|R| - 1)$
  - Apply induction hypothesis to $S(P')$, $S(P'')$ $\square$

- Explorer meets visited cell c'
- R: shortest path from c to c'



If there is a split cell in the first layer, we have two cases. In the first case, the new component was never visited before, and we define R to be the split cell.
In the second case, we meet another cell, c', and let R enclose the shortest path from c to c'.
Now, we split our polygon into P' and P''. We have this equation for the number of cells, because we count the cells in R twice, and also this equation for the number of steps, because we count the steps between c and c' twice.
Applying the induction hypothesis to the number of steps in P' and P'' yields our result.

Problem: Online exploration of simple grid polygons

- Lower bound △: $\frac{7}{6}$
- Lower bound ⬡: $\frac{14}{13}$
- Exploration strategy *SmartDFS*
- Upper bound △: $S(P) \leq C(P) + E(P) - 4$
- Upper bound ⬡: $S(P) \leq C(P) + \frac{1}{4}E(P) - \frac{5}{2}$
- $\frac{4}{3}$-competitive

- ToDo: Close the gap!

Thank you!