A 4-Approximation Algorithm for Guarding 1.5D Terrains

James King
McGill University

March 21, 2006
Outline

- Introduction to Terrain Guarding
- Previous Work
- Preliminaries
- Our 4-Approximation Algorithm
- Future Work
What is a 1.5D Terrain?

▶ Also known as an $x$-monotone chain.
▶ The terrain intersects any vertical line at most once.
▶ No caves or overhangs.

▶ Points on the terrain ‘see’ each other if the line segment connecting them is never below the terrain.
What is a 1.5D Terrain?

- Also known as an $x$-monotone chain.
- The terrain intersects any vertical line at most once.
- No caves or overhangs.

- Points on the terrain ‘see’ each other if the line segment connecting them is never below the terrain.
The Terrain Guarding Problem

- We want a minimum set of guards that see the entire terrain.
- This is very similar to the *Art Gallery Problem*.
The Terrain Guarding Problem

The **continuous** problem:
- The entire terrain must be guarded.
- Guards can be placed anywhere.
- Closer to real-life applications.

The **discrete** problem:
- Only vertices need to be guarded.
- Guards can only be placed on vertices.
- Closely related to non-geometric problems (e.g. Vertex Cover).
Outline

Introduction to Terrain Guarding

Previous Work

Preliminaries

Our 4-Approximation Algorithm

Future Work
Is 1.5D Terrain Guarding NP-Complete?

- We don’t know.
- Many related problems are NP-complete:
  - Art Gallery Problem.
  - Vertex Domination.
  - Set Cover.
  - 2.5D Terrain Guarding.
- However, 1.5D Terrains seem to forbid complex constructions.
Is 1.5D Terrain Guarding NP-Complete?

- We don’t know.
- Many related problems are NP-complete:
  - Art Gallery Problem.
  - Vertex Domination.
  - Set Cover.
  - 2.5D Terrain Guarding.
- However, 1.5D Terrains seem to forbid complex constructions.
Terrain Guarding Algorithms

- Constant factor approximation algorithms given by:
  - Ben-Moshe, Katz, and Mitchell.
  - Clarkson and Varadarajan (randomized).
- They did not attempt to minimize the approximation factor.
The Terrain Guarding Problem

- Our contribution is a 4-approximation algorithm.
- Simpler than previous algorithms.
- Best approximation factor so far.
- We will present the algorithm for the \textbf{discrete} problem.
- Works for the continuous problem with a slight modification.
- Runs in $\Theta(n^2)$ time.
Outline

Introduction to Terrain Guarding
Previous Work
Preliminaries
Our 4-Approximation Algorithm
Future Work
Terminology and Notation

- $p < q$ means $p$ is to the left of $q$.
- $p$ dominates $q$ means $p$ sees every unguarded point that $q$ sees.
- $L(p)$ is the leftmost point that sees $p$.
- $R(p)$ is the rightmost point that sees $p$.
Order Claim

- The fundamental property of 1.5D terrains that we exploit.

- Consider $a < b < c < d$.
- If $a$ sees $c$ and $b$ sees $d$ then $a$ sees $d$. 
External Domination

- Consider $a < b < c$ such that $a$ sees $c$.
- $\{a, c\}$ dominates $b$ outside the interval $(a, c)$. 
The Algorithm

while unguarded points remain do
    find $u$, $S(u)$ such that:
    • $u$ is unguarded
    • $|S(u)| \leq 4$
    • $S(u)$ dominates any guard that sees $u$
    place guards at the points in $S(u)$
end while

▶ This guarantees an approximation factor of 4.
▶ The real work is finding $u$ and $S(u)$. 
Finding \textit{u} and \textit{S(u)}

- The work is done by \texttt{GuardRight}, a recursive method.
- \texttt{GuardLeft} is the mirror image of \texttt{GuardRight}.

\texttt{GuardRight} recurses by calling \texttt{GuardLeft} and vice versa.

Guards are placed only in the terminal case.
\textbf{GuardRight}(p, c)

- \(p\) is an unguarded vertex (not on the convex hull).
- \(c\) sees every unguarded vertex in \([L(R(p)), p)\)
- We look for \(u\) and \(S(u)\) in \([L(R(p)), R(p)]\).
- We either find them or isolate a subregion to recurse on.
Guards on the Left

- Left vertices (●) are on the convex hull between $p$ and $L(R(p))$.
- Right vertices (●) are between $p$ and $R(p)$.
- If we place a guard at $c$ we don’t need to consider placing guards in $[L(R(p)), p)$ except on left vertices.
Yet More Terminology

- Unguarded vertices in \([p, R(p))\) are either exposed or sheltered.
- Exposed vertices (○) can see a left vertex.
- Sheltered vertices (●) cannot see any left vertex.
- \(p\) is exposed.
- \(d\) is a special exposed vertex that our algorithm finds.
The Terminal Case

- $L(d)$ sees every exposed vertex to the right of $L'(d)$.
  - $L'(d)$ is the leftmost right vertex that sees $d$.
- If $\{c, L(d), L'(d), R(d)\}$ sees every unguarded vertex in $[L'(d), R(d)]$ then the set dominates any vertex that sees $d$.
- This is the terminal case so we place guards.

- $u \leftarrow d$.
- $S(u) \leftarrow \{c, L(d), L'(d), R(d)\}$. 
The Recursive Case

- If not terminal, there must be a sheltered vertex in \([L'(d), R(p)]\).
- Define \(q\) as the rightmost sheltered vertex.
- \(L(d)\) sees every unguarded vertex in \((q, R(L(q))]\).
- We call \texttt{GuardLeft}(q, L(d)).
The Recursive Case

- We recurse on $[L(q), R(L(q))]$, which is a proper subterrain of $[L(R(p)), R(p)]$.
- Problem size shrinks, so we reach a terminal case eventually.
- We just repeat this whole process until the entire terrain is guarded.
Outline

- Introduction to Terrain Guarding
- Previous Work
- Preliminaries
- Our 4-Approximation Algorithm
- Future Work
Future Work

- Primary question: is 1.5D terrain guarding in P or is it NP-complete?
- Characterize visibility graphs of terrains.
  - The order claim isn’t the only tool we can use!
  - What other tools are available?
Thank you!