Theoretical Aspects of Intruder Search
Course Wintersemester 2015/16
Graphs and Trees

Elmar Langetepe

University of Bonn

October 20th, 2015
Organisation

- Lecture: Tuesday 16:15 to 17:45
- Exercise groups: Starting 28/29th
  - Wednesday: 14:15-15:45
  - Thursday: 10:15-11:45
- Sign in
- Manuscript on the webpage
- Slides on the webpage
- Exercises
- Today: Introduction
Repetition: Main problems and intention

- Evader/Intruder versus Searcher/Guard
- Escaping/Intruding versus Catching/Avoidance
- Game, Competition
- Different Scenarios: Environment, Facilities, Goal, Model
- Discrete, Continuous, Geometry, Combinatorics
- Interpretation: Possible Position of the Intruder, Decontamination, Firefighting
Repetition: Theoretical Aspects

- Algorithmic track
- Computational complexity
- Correctness or Failure
- Efficiency
- Optimality
- Prerequisites: Algorithms, Datastructure, Analysis, Complexity, Computability
- Models, Methods, Proof Techniques, Tools
Repetition: Different Examples

1. Optimal-Closing-Sequence against Intruder
   - Saving maximum area by closing doors in polygon
   - NP-hard, Reduction of Subset-Sum

2. Catching-an-Evader
   - Grid-Graph game, Evader moves, $k$ stationary Guards
   - Correctness $k = 1$ impossible, $k = 2$, optimal solution by ILP

3. Enclosing-a-Fire
   - Expanding circle in the plane, Build a barrier with speed $v > 1$
   - Barrier Curves: Circle around origin of fire, $v \geq 2\pi$ tight

4. Discrete Fire-Figthing-Curve
   - Grid-Graph, Fire spreads after $n$ steps, barrier cell after $b$ steps
   - Conjecture, $b < \frac{n-1}{2}$ tight bound! Simulation!
Chapter 2: Discrete Scenarios

- Graph $G = (V, E)$, degree $d$, root $r$, $p$ firefighters per step
- Different models: Intruder-Search/Firefigthing
- Complexity
- Optimal Algorithms
- Approximation
Graph of degree 3

- NP-complete
- Simple Algorithm for root vertex of degree 2
- Defending a vertex
- $\text{dist}(u, r)$ length of a shortest path from $r$ to $u$
- $V_1$ vertices of degree 1, $V_2$ vertices of degree 2
- $V_c$ vertices of degree 3 that belong to a cycle
- Example
Graph of degree 3, root vertex degree 2

Path Strategy

**Lemma 6**: Vertex $u \in V_1 \cup V_2$ can be enclosed in time $\text{dist}(u, r) + 1$ and only $\text{dist}(u, r) + 1$ vertices are on fire. Vertex $u \in V_c$ can be enclosed in time $\text{dist}(u, r) + C(u) - 1$ and only $\text{dist}(u, r) + C(u) - 1$ vertices are on fire.

Proof! Constructive!
Optimal Strategy:

\[ f(u) := \begin{cases} 
\text{dist}(u, r) + 1 & : \text{if } u \in V_1 \cup V_2 \\
\text{dist}(u, r) + C(u) - 1 & : \text{if } u \in V_c \setminus V_2 \\
\infty & : \text{otherwise}
\end{cases} \]

Find a vertex \( u \) with \( f(u) = \min_{x \in V} f(x) \). Enclose this vertex by the path strategy.
Structural Property

**Lemma 7:** For a setting \((G, r, 1)\) where \(G\) has maximal degree 3 and root \(r\) has degree \(\leq 2\) there is always an optimal protection strategy that protects the neighbor of a contaminated vertex in each time step.

Proof!

- Trivial for degree \(r\) is 1
- Degree of \(r\) is 2, minimal counterexample
- I. one of the neighbors of \(r\) first, II. not a neighbor of \(r\)
Optimal Strategy:

\[
f(u) := \begin{cases} 
\text{dist}(u, r) + 1 & : \text{if } u \in V_1 \cup V_2 \\
\text{dist}(u, r) + C(u) - 1 & : \text{if } u \in V_c \setminus V_2 \\
\infty & : \text{otherwise}
\end{cases}
\]

Find a vertex \( u \) with \( f(u) = \min_{x \in V} f(x) \). Enclose this vertex by the path strategy.

**Theorem 8:** For a problem instance \((G, r, 1)\) of a graph \( G \) of maximum degree 3 and a root vertex of degree 2 the above strategy is optimal.

Proof!
Theorem 8: For a problem instance \((G, r, 1)\) of a graph \(G\) of maximum degree 3 and a root vertex of degree 2 the above strategy is optimal.

Proof:

- Last burning vertex \(u\)
- \(u \in V_1, V_2\), by construction
- \(u \in V_c\), neighbors \(n_1, n_2, n_3\), and \(n_1\) on fire
- Also \(n_2\) on fire: \(u, n_1\) and \(n_2\) on a cycle, contradiction!
- \(n_2\) and \(n_3\) are protected.
- Another neighbor of \(n_2\) or \(n_3\) is on fire, say \(p\) of \(n_2\)
- Otherwise: protect \(u\) one step earlier
- \(u, n_2, p\) build the cycle
**Theorem 9:** For a problem instance \((G, r, 1)\) of a graph \(G = (V, E)\) of maximum degree 3 and a root vertex of degree 2 the decision problem can be solved in polynomial time and the maximum number of vertices that can be saved is \(|V| - \min_{x \in V} f(x)\).

Proof: Compute the values in polynomial time!
Theorem 10: The firefighter decision problem for graphs is NP-hard.

Proof:

- k-Clique reduction to the decision problem
- Construct Graph $G' = (V', E')$ from Graph $G = (V, E)$
- Vertex $s_v$ for every $v \in V$, Vertex $s_e$ for every $e \in E$
- Edges $(s_v, s_e)$ and $(s_u, s_e)$ for every edge $e = (u, v)$
- Root vertex $r$ and $k - 1$ columns of $k$ vertices $v_{i,j}$
- Connect the layer from left to right and to all $s_v$
- Example Blackboard!
Theorem 10: The firefighter decision problem for graphs is NP-hard.

Proof:

1. k-Clique exists
   - Save $k$ vertices of the $k$ Clique
   - Saves $k' = k + \left( \frac{k}{2} \right) + 1$ vertices

2. Saving $k' = k + \left( \frac{k}{2} \right) + 1$ vertices?
   - Before step $k$: Saving $s_e$ or $a_{i,j}$ needless, only one
   - Saving $k$ vertices $s_v$.
   - $k'$ possible, if $k$-Clique exists
   - Another one in the last step
Theorem 11: For a problem instance \((T, r, 1)\) of a rooted tree \(T = (V, E)\) the greedy strategy gives a \(\frac{1}{2}\) approximation for the optimal number of vertices protected. This bound is tight.

Proof:

1. Example for \(\frac{k+1}{2(k-1)} \mapsto \frac{1}{2}\)

2. Tightness
   - Greedy versus opt, time steps: Savings
   - \(\text{opt}_A\) not better than greedy, \(\text{opt}_B\) better than greedy.
   - \(2S_G \geq \text{opt}_A + \text{opt}_B\)
   - Greedy competes with \(\text{opt}_A\) at the start
   - Moment where Greedy is worse than \(\text{opt}\)
   - \(\text{opt}_B\) choose \(v\), depth \(l\), also greedy can choose \(l\) or greedy has chosen a predecessor of \(v\) before \(\Rightarrow\) greedy saves at least the vertices of \(\text{opt}_B\) before
Efficient Algorithm for Trees

Firefighter Decision Problem (Protection by k guards):

**Instance:** A Graph $G = (V, E)$ of degree $d$ with root vertex $r$ and $p$ firefigther per step and an integer $k$.

**Question:** What is the strategy that saves a maximum number of vertices by protecting $k$ vertices in total?
Dynamic Programming Approach: Place $k$ guards! Structural Property!

**Lemma 12:** For any optimal strategy for an instance of the firefighter decision problems on trees (protection by $k$ guards, saving $k$ vertices) the vertex defended at each time is adjacent to a burning vertex. There is an integer $l$, so that all protected vertices have depth at most $l$, exactly one vertex $p_i$ at each depth is protected and all ancestors of $p_i$ are burning.

- Time step $t$, place guard with non-burning neighbor
- Placement closer to the root improves strategy
- Depth $t$ at step $t$, inductively!
Efficient Algorithm for Trees

Dynamic Programming Approach: Place $k$ guards!

- Lemma 12: Guards in depth 1, 2, \ldots, $k$
- $L_k$, vertices of $T$ with depth $\leq k$
- Order for the processing: Subproblems!
- Preorder of the graph! *to the left, rightmost*
- $l(v)$, vertex to the left of $v$
- $T_v$ subtree at $v$
- $T^v$ tree with vertices from $L_k$ to the left of $v$, including $v$
- Recursion more general: Vector $X \in \{1, 0\}^k$
- $X(j) = 1$ place guard in step $j$, $X(j) = 0$ n guard in step $j$
- $A_v(X)$: Optimal strategy for $X$ in $T^v$, based on $T^{l(v)}$
- Recursion!
Problem! No two guards on a single path! Set guard after depth $i$!
\[ A_v(X, i) := \]

Optimal protection number in \( T^v \) for strategy that sets the guards w.r.t. entries of \( X \) and no guard is set on the path from \( r \) to \( v \) at depth \( \leq i \)
**Theorem 13:** Computing the optimal protection strategy for $k$ guards on a tree $T$ of size $n$ can be done in $O(n2^k k)$ time.

\[
A_v(X, i) := \max \begin{cases} 
A_{l(v)}(X, \min(d(v) - 1, i)) \\
[X(d(v)) = 1 & d(v) > i] \cdot (|T_v| + A_{l(v)}(X^v, d(v) - 1)) 
\end{cases}
\]

- Compute $L_k$, $l(v)$, $|T^v|$ in linear time!
- Traverse the vertices of $L_k$ from left to right
- At most $n \times 2^k \times k$ entries $A_v(X, i)$
- $n$ stands for $v$, $2^k$ stands for $X$, $k$ stands for $i$. 
Corollary 14: Computing a strategy for a tree $T$ of size $n$ that saves at least $k$ vertices can be done in $O(n2^k k)$ time.

- Run above algorithm for $i = 1, \ldots, k$
- Sufficient!
- $\sum_{i=1}^{k} i2^i n \leq kn \sum_{i=1}^{k} 2^i = (2^{k+1} - 2)kn$