Theoretical Aspects of Intruder Search
Course Wintersemester 2015/16
Dynamic strategies on Trees

Elmar Langetepe

University of Bonn

November 10th, 2015
Start vertex $v$ and order of the subtrees:

$$\text{cs}(T_v(z)) = \max\{\text{cs}(T_v(z_1)), \text{cs}(T_v(z_2)) + w(z)\}$$
Startvertex $v$ and order of the subtrees:

\[ cs(T_v(z)) = \max\{cs(T_v(z_1)), cs(T_v(z_2)) + w(z)\} \]
Lemma 23: Let $z_1, \ldots, z_d$ be the $d \geq 2$ children of a vertex $z$ in $T_v$ and assume that $cs(T_v(z_i)) \geq cs(T_v(z_{i+1}))$ for $i = 1, \ldots, d - 1$. We have

$$cs(T_v(z)) = \max\{cs(T_v(z_1)), cs(T_v(z_2)) + w(z)\} \quad (1)$$

if the tree $T$ is a tree with unit weights.

Proof:

- $cs(T_v(z)) \geq cs(T_v(z_1))$, order of cleaning
- Case 1: $cs(T_v(z_1)) \geq cs(T_v(z_2)) + w(z)$
- Clear $T_v(z)$, set $w(z)$ on $z$, clear all $T_v(z_i)$ by $cs(T_v(z_1))$ agents but $T_v(z_1)$ last
- Case 2: $cs(T_v(z_1)) < cs(T_v(z_2)) + w(z)$ is necessary!
Lemma 23: Let $z_1, \ldots, z_d$ be the $d \geq 2$ children of a vertex $z$ in $T_v$ and assume that $cs(T_v(z_i)) \geq cs(T_v(z_{i+1}))$ for $i = 1, \ldots, d - 1$. We have

$$cs(T_v(z)) = \max\{cs(T_v(z_1)), cs(T_v(z_2)) + w(z)\}$$

(2)

if the tree $T$ is a tree with unit weights.

Case 2: $cs(T_v(z_1)) < cs(T_v(z_2)) + w(z)$

Show: $cs(T_v(z_2)) + w(z) - 1$ not sufficient

1. $T_v(z_2)$ is cleared before $T_v(z_1)$: While $cs(T_v(z_2))$ agents clear $T_v(z_2)$ there are only $w(z) - 1 = 0$ agents left for blocking a vertex in $T_v(z_1)$. Recontamination!

2. $T_v(z_1)$ is cleared before $T_v(z_2)$: While $cs(T_v(z_1))$ agents clear $T_v(z_1)$ there are no more $w(z) - 1 = 0$ agents left for blocking a vertex in $T_v(z_2)$ (because $cs(T_v(z_1)) = cs(T_v(z_2))$). Recontamination!
Design of a strategy: Example! Barriere et al. Flaw!

\[ cs(T_v(z)) = \max \{ cs(T_v(z_1)), cs(T_v(z_2)) + w(z) \} \] (3)

\[ \max \{ cs(T_x(z_1)), cs(T_x(z_2)) + w(v) \} = \max \{ 8, 7 + 5 \} = 12 \]

But 10 agents are also sufficient!
Corollary 24: For a unit weighted tree $T$ of size $n$ and for a given starting vertex $v$ we can compute the optimal monotone contiguous strategy starting at $v$ in $O(n)$ time. An overall optimal contiguous strategy can be computed in $O(n^2)$.

Proof: For any root $v$ compute the values $cs(T_v(x))$ starting from the leafes. Do this for all $v \in T$. 
Labels in the tree

Compute the information in one walkthrough!
Local recursive labeling: $\lambda_x(e)$ for the links $e = (x, y)$ adjacent to $x$.

Let $e = (x, y)$ be a link incident to $x$.

1. If $y$ is a leaf, set $\lambda_x(e) = w(y)$.

2. Otherwise, let $d$ be the degree of $y$ and let $x_1, \ldots, x_{d-1}$ be the incident vertices of $y$ different form $x$. Let $\lambda_y(y, x_i) =: l_i$ and $l_i \geq l_{i+1}$. Then,

$$\lambda_x(e) := \max\{l_1, l_2 + w(y)\}.$$
1 Start with the leaves and for any leaf $y$ and for $e = (x, y)$ send a message $l = w(y)$ to $x$. After receiving this messages, $x$ sets $\lambda_x(e) = l$.

2 Consider a vertex $y$ of degree $d$ that has received at least $d - 1$ messages $l_i$ from the incident certices $x_1, \ldots, x_{d-1}$ and let $x$ be the remaining incident vertex. Let $l_i \geq l_{i+1}$. Send a message $l = \max\{l_1, l_2 + w(y)\}$ to $x$, after receiving the message $x$, set $\lambda_x((x, y)) = l$. 
Example for general tree

$\lambda_{v_3}(e_1) = 7$
$\lambda_{v_4}(e_1) = 10$
$\lambda_{v_5}(e_4) = 6$
$\lambda_{v_5}(e_5) = 1$
$\lambda_{v_5}(e_6) = 1$
$\lambda_{v_5}(e_7) = 1$
$\lambda_{v_6}(e_5) = 4$
$\lambda_{v_6}(e_6) = 10$

$\lambda_{v_1}(e_2) = 3$
$\lambda_{v_1}(e_3) = 5$
$\lambda_{v_2}(e_3) = 10$
$\lambda_{v_2}(e_4) = 6$
$\lambda_{v_3}(e_1) = 7$
$\lambda_{v_3}(e_2) = 3$
$\lambda_{v_3}(e_4) = 10$
$\lambda_{v_3}(e_5) = 6$
$\lambda_{v_3}(e_6) = 10$
$\lambda_{v_3}(e_7) = 12$

$v_1, v_2, v_3, v_4, v_5, v_6, v_7$
Lemma 24: The links of a tree $T$ can be labeled with labels $\lambda_x$ by the above message sending algorithm by $O(n)$ messages in total.

Proof by construction!
Lemma 26: For a unit weighted tree $T = (V, E)$ and an edge $e = (x, y) \in E$ we have $cs(T_x(y)) = \lambda_x(e)$.

Proof: By induction!

- $y$ leaf and $\lambda_x(e) = w(y)$ for $h(y) = 0$
- Statement holds for $0 \leq h(y) < k$ and consider $h(y) = k$
- $e = (x, y), x_1, \ldots, x_d$ the $d \geq 1$ children of $y$ in $T_x(y)$
- $T_y(x_i) = \lambda_y((y, x_i))$ by induction hypothesis, $T_y(x_i) = T_x(x_i)$ by definition
- $cs(T_x(x_i)) \geq cs(T_x(x_{i+1}))$ for $i = 1, \ldots, d - 1$
- Recursion for $T_x(y)$ and $\lambda_x((x, y))$ identical!
Order all $\lambda_v((v, x_i))$ for all $i = 1, \ldots, d$ incident edges $(v, x_i)$ so that $\lambda_v((v, x_i)) \geq \lambda_v((v, x_{i+1}))$, compute

$$\mu(v) = \max\{\lambda_v((v, x_1)), \lambda_v((v, x_2)) + w(v)\}.$$  

(4)

$$\mu(v) = \text{cs}(T_v) \text{ and } \min_{v \in V} \mu(v) = \text{cs}(T).$$

Strategy: By the increasing order of the values $\lambda_x$ at vertex $x$!
\[ \mu(v_3) = \max(\lambda_{v_3}(e_1), \lambda_{v_3}(e_3) + 7) = 12 \]
\[ \mu(v_5) = \max(\lambda_{v_5}(e_4), \lambda_{v_5}(e_5) + 5) = 10 \]
\[ 10.\lambda_{v_7}(e_6) = 10 \]

8. \( \lambda_{v_3}(e_1) = 7 \)
4. \( \lambda_{v_4}(e_1) = 10 \)
6. \( \lambda_{v_4}(e_4) = 6 \)
3. \( \lambda_{v_5}(e_6) = 1 \)
1. \( \lambda_{v_5}(e_5) = 1 \)

7. \( \lambda_{v_5}(e_4) = 10 \)
5. \( \lambda_{v_5}(e_5) = 4 \)
4. \( \lambda_{v_6}(e_5) = 10 \)

2. \( \lambda_{v_3}(e_3) = 5 \)
1. \( \lambda_{v_3}(e_2) = 3 \)
11. \( \lambda_{v_2}(e_3) = 10 \)
12. \( \lambda_{v_1}(e_2) = 12 \)

\[ \mu(v_3) = \max(\lambda_{v_3}(e_1), \lambda_{v_3}(e_3) + 7) = 12 \]
\[ \mu(v_5) = \max(\lambda_{v_5}(e_4), \lambda_{v_5}(e_5) + 5) = 10 \]
\[ 10.\lambda_{v_7}(e_6) = 10 \]

8. \( \lambda_{v_3}(e_1) = 7 \)
4. \( \lambda_{v_4}(e_1) = 10 \)
6. \( \lambda_{v_4}(e_4) = 6 \)
3. \( \lambda_{v_5}(e_6) = 1 \)
1. \( \lambda_{v_5}(e_5) = 1 \)

7. \( \lambda_{v_5}(e_4) = 10 \)
5. \( \lambda_{v_5}(e_5) = 4 \)
4. \( \lambda_{v_6}(e_5) = 10 \)

2. \( \lambda_{v_3}(e_3) = 5 \)
1. \( \lambda_{v_3}(e_2) = 3 \)
11. \( \lambda_{v_2}(e_3) = 10 \)
12. \( \lambda_{v_1}(e_2) = 12 \)
Theorem 27: On optimal contiguous strategy for a unit weighted tree $T = (V, E)$ can be computed in $O(n)$ time and space.

Proof:

- Calc. messages an $\mu$ values in $O(n)$ time
- Register only three greatest values for every vertex

Example: Applet!
Theorem 28: For unit weights and for any number of vertices $n$, we have $\lfloor \log_2 n \rfloor - 1 \leq cs(n) \leq \lfloor \log_2 n \rfloor$.

Two directions!
Lemma 29: For every \( n \geq 1 \) we find trees \( T_n \) with
\[
\text{cs}(T_n) \geq \lceil \log_2(\frac{2}{3}(n + 1)) \rceil \geq \lceil \log_2 n \rceil - 1.
\]

Proof:

- Case 1: \( n \) equals \( 2^k - 1 \)
  - Choose complete binary tree
  - \( \text{cs}(T_n) = k - 1 = \log_2(n + 1) - 1 \geq \log_2 \lceil \frac{2}{3}(n + 1) \rceil \)
Lower and upper bounds for the contiguous search

- Case 1: $n$ equals $2^k - 1$
- $cs(T_n) = k - 1 = \log_2(n + 1) - 1 \geq \log_2\left(\frac{2}{3}(n + 1)\right)$

$k = 4$ and $n = 2^k - 1$

$\lambda_v((v, u)) = k - \text{level}(u)$
$\lambda_u((v, u)) = k - 1$
$\mu(r) = k$ and $\mu(u \neq r) = k - 1$
Lemma 29: For every \( n \geq 1 \) we find trees \( T_n \) with
\[
cs(T_n) \geq \lfloor \log_2 \left( \frac{2}{3} (n + 1) \right) \rfloor \geq \lfloor \log_2 n \rfloor - 1.
\]

Proof:

- Case 1: \( n \) equals \( 2^k - 1 \)
- Case 2: \( n \) does not equal \( 2^k - 1 \)
- \( n = \sum_{i=1}^{r} 2^{\alpha_i} \) with \( \alpha_1 > \alpha_2 > \cdots > \alpha_r \).
- \( n = 11010 \) in binary representation with \( \alpha_1 = 4, \alpha_2 = 3, \alpha_3 = 2 \).
- Chain of vertices \( x_1, x_2, \ldots, x_r \)
- For any \( x_i \) connect complete binary tree \( T_{\alpha_i} \) of size \( 2^{\alpha_i} - 1 \)
- \( 2^{\alpha_1} - 1 < n < 2^{\alpha_1+1} - 1 \) and require
  \[
  cs(T_n) = \alpha_1 \geq \log_2 (n + 1) - 1 \geq \log_2 \left( \frac{2}{3} (n + 1) \right) \]
Case 2: \( n \) does not equal \( 2^k - 1 \)

\[
\text{cs}(T_n) = \alpha_1 \geq \log_2(n + 1) - 1 \geq \log_2\left(\frac{2}{3}(n + 1)\right)
\]

\[
n = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 11010
\]

\[
\lambda_{y_1}((v, y_1)) = \alpha_1 - 1
\]

\[
\lambda_{y_1}((x_1, y_1)) = \alpha_2 + 1 = \alpha_1
\]
Lemma 30: For every $n \geq 1$ and unit weights, $\lfloor \log_2 n \rfloor$ agents are sufficient for a contiguous search strategy.

Proof: Arbitrary tree $T_r$ with root $r$, $cs(T)$, construct $T_r'$

1. For a node $x$ and its $d > 2$ children $x_1, x_2, \ldots, x_d$ ordered by $cs(T_r(x_i)) \geq cs(T_r(x_{i+1}))$ remove all $T_r(x_i)$ for $i > 2$.

2. For a node $x$ with two children $x_1$ and $x_2$ and $cs(T_r(x_1)) > cs(T_r(x_2))$, remove $T_r(x_2)$.

3. For a node $x \neq r$ with only one child $x_1$, remove $x$ and connect $x_1$ to the parent of $x$.

4. If there are more than two vertices left, and $r$ has only one child $x_1$, remove $x_1$ and connect the children of $x_1$ to $r$. 
Lemma 30: For every \( n \geq 1 \) and unit weights, \( \lfloor \log_2 n \rfloor \) agents are sufficient for a contiguous search strategy.

Proof:

- Agents required for \( T \) and \( T_r \) are the same, computation of \( \mu(r) \) in \( T_r \) use the same values.
- Weights restricted to one, rule 2. is correct by \( \text{cs}(T_r(x_1)) \geq \text{cs}(T_r(x_2)) + 1 \).
Lower and upper bounds for the contiguous search

1. Binary: Any inner vertex has no more than 2 children! Rule 1 and 2!

   Rule three deletes internal nodes with one child except for the root. Rule 4 make the root have 2 or 0 children.

1. For a node $x$ and its $d > 2$ children $x_1, x_2, \ldots, x_d$ ordered by $cs(T_r(x_i)) \geq cs(T_r(x_{i+1}))$ remove all $T_r(x_i)$ for $i > 2$.

2. For a node $x$ with two children $x_1$ and $x_2$ and $cs(T_r(x_1)) > cs(T_r(x_2))$, remove $T_r(x_2)$.

3. For a node $x \neq r$ with only one child $x_1$, remove $x$ and connect $x_1$ to the parent of $x$.

4. If there are more than two vertices left, and $r$ has only one child $x_1$, remove $x_1$ and connect the children of $x_1$ to $r$. 
1. Complete: $T'_x$ not complete and no subtree in $T'_x$ incomplete

1. For a node $x$ and its $d > 2$ children $x_1, x_2, \ldots, x_d$ ordered by $\text{cs}(T_r(x_i)) \geq \text{cs}(T_r(x_{i+1}))$ remove all $T_r(x_i)$ for $i > 2$.

2. For a node $x$ with two children $x_1$ and $x_2$ and $\text{cs}(T_r(x_1)) > \text{cs}(T_r(x_2))$, remove $T_r(x_2)$.

3. For a node $x \neq r$ with only one child $x_1$, remove $x$ and connect $x_1$ to the parent of $x$.

4. If there are more than two vertices left, and $r$ has only one child $x_1$, remove $x_1$ and connect the children of $x_1$ to $r$. 