Solving Pursuit-Evasion Problems with Graph-Clear: an Overview

Andreas Kolling\textsuperscript{1}    Stefano Carpin\textsuperscript{2}

\textsuperscript{1}School of Information Sciences
University of Pittsburgh, USA

\textsuperscript{2}School of Engineering
University of California, Merced, USA

http://robotics.ucmerced.edu
Outline

1. The Graph Clear problem
   - Motivation
   - Model definition
   - Theoretical properties

2. Algorithms for Graph Clear on trees
   - Deterministic case
   - Probabilistic case

3. Line Clear
   - Motivation
   - Dealing with unknown environments

4. Conclusions
Looking for intruders: environment
Looking for intruders: contamination
Looking for intruders: pursuers
Hypotheses

- bounded multiply-connected environments possibly hiding an unknown number of intruders
- multiple robots with limited sensing capabilities
- intruders are *eliminated* as soon as they fell into the sensing range of at least one pursuer
- intruders move at *unbounded speed* and have complete knowledge of pursuers’ positions
- intruders follow continuous paths (no jumping allowed)
Graph abstraction

Motivation: topological maps used in robotics
Edge search

T. Parsons, *Pursuit-evasion in a graph*, 1976
L. Barriere et al. *Capture of an intruder by mobile agents*, 2002
Removing contamination: clearing and blocking

- robots may remove contamination applying two operations:
  - **clear**: remove all contamination from a vertex
  - **block**: prevents an intruder from passing unobserved through an edge
- an edge is **blocked** if a block operation is applied
- a cleared vertex/edge becomes **recontaminated** if there exist a path to a contaminated vertex/edge
- intruders move at unbounded speed and have full knowledge of the pursuers’ strategy
  ⇒ recontamination happens as soon as it is possible
Costs

Robots have limited sensing capabilities:

- blocking an edge may need more than one robot
  ⇒ cost of a block is the weight of a vertex $w(v)$
- clearing a vertex may need more than one robot
  ⇒ cost of clearing is the weight of an edge $w(e)$
- Hypothesis: all edges emanating from a vertex must be blocked while clearing

$$s(v) = w(v) + \sum_{e_j \in \text{edges}(v)} w(e_j)$$
Surveillance graph

Definition

A *surveillance graph* is a triple $G = (V, E, w)$, where $(V, E)$ is an undirected graph with vertex set $V$, edge set $E$, and $w : V \cup E \rightarrow \mathbb{N}^+$ as a weight function. Vertices and edges in a surveillance graph have a state. The state of a vertex can be *clear*, or *contaminated*, while the state of an edge can be *clear*, *contaminated* or *blocked*. 
Strategy

- A sequence of coordinated block and clear actions that turn a fully contaminated graph into a fully cleared graph
  - should consider recontamination
  - intruders may even know the strategy and take advantage from its weaknesses
- Coordination is needed to
  - clear a vertex enforcing the blocking constraints
  - block formerly cleared edges to prevent recontamination
- **Cost of a strategy**: minimum number of robots needed to implement it
- **A contiguous strategy** requires that cleared parts always form a connected subgraph of $G$
An example of strategy
An example of strategy

Cost = 5
An example of strategy

Cost = 6
An example of strategy

Cost = 8
An example of strategy

Cost = 6
An example of strategy

Cost = 11
An example of strategy

Cost = 7
An example of strategy

Cost = 6
The Graph Clear problem

Definition (Graph Clear problem)
Let $G$ be a fully contaminated surveillance graph. Determine a strategy $S$ for $G$ of minimal cost.

Definition (Cost of a graph)
Let $G$ be a surveillance graph and let $S$ be a strategy of minimal cost for $G$. We define the cost of graph $G$ as $ag(G) = ag(S)$. 
Multiple blocks don’t help

Lemma

Let $S = \{a_1, \ldots, a_k\}$ be a strategy for $G$. Then there exists a strategy $S'$ with cost $ag(S') \leq ag(S)$ that sweeps no more than a vertex at the time.
Decision version of the Graph Clear problem:

INSTANCE: $G = (V, E, w)$, a surveillance graph with $w(x) = 1 \ \forall x \in V \cup E$, and a natural number $P$

QUESTION: is $ag(G) \leq P$?
The complexity of GRAPH-CLEAR

Theorem

*The decision version of GRAPH CLEAR is NP-complete.*

Proof by duality from Megiddo's theorem.

Theorem

*Recontamination does not help for contiguous strategies on trees.*

Label based algorithms

Let \( e = [v_x, v_y] \) be an edge. Define two labels, \( \lambda_{v_x}(e) \) and \( \lambda_{v_y}(e) \):

- \( \lambda_{v_x}(e) \): number of agents needed to clear the subtree rooted in \( v_y \) when coming from \( v_x \)

![Diagram showing a tree with nodes v_x and v_y connected by edge e]
Two algorithms for contiguous strategies

- $O(n \log d)$ easy to implement algorithm. But produces in general suboptimal strategies.
  - sometimes fully clearing a subtree is not convenient


- $O(n^2)$ provably optimal algorithm.

Non-contiguous strategies

- less constrained and in general better
- hybrid-strategies: clear certain subtrees contiguously and certain non-contiguously
- dynamic programming based approach runs in pseudopolynomial time
- Open question: can we find optimal non-contiguous strategies?


Accounting for imprecise sensors

Assume sensors are subject to false negatives with a certain probability:

- a strategy is executed and no detection is reported. What is the probability that $n$ intruders managed to stay undetected?
- given $r$ robots, how should they cooperate to minimize the failure rate?
- how many robots to I need to ensure that if no detections are reported than with probability at least $p$ no intruders are present?
Extending the model

Weight function: \( w : V \cup E \rightarrow \mathcal{F} \)

\[
\mathcal{F} = \{ f \mid f : \mathbb{N} \rightarrow [0, 1], f(0) = 1, \forall r, r' \in \mathbb{N} \ r \geq r' \Rightarrow f(r) \leq f(r') \}
\]

\[
\bar{w}_x = \min \{ n \mid w_x(n) < 1 \}
\]

\[
w_x(r) = p(\bar{N}|t, r)
\]
Probabilistic strategies

Need to specify not only where to block/clear, but also how many robots to use.

- discrete random variable $T$: number of undetected intruders with known prior

$$p(T = i | \bar{N}) = \frac{p(\bar{N} | T = i) \cdot p(T = i)}{p(\bar{N})}$$

Goal: compute $p(\bar{N} | T = i)$ starting from $w_x$

Worst case adversary:

- crosses exactly once from contaminated back into cleared where fault rate is maximized
- independently detected

$$p(\bar{N} | T = i) = p_{x_{\text{max}}} (N | t)^i$$

$$x_{\text{max}} = \text{argmax}_{x \in G} \{w_x\}$$
Modeling faulty sensors

\[ p_e(N \mid t) = \prod_{i=1}^{n_p} p_{g_i}(N \mid t) \]
Combining failure rates

- for each edge $e$ compute a failure function rather than a label $\lambda_{v_x}(e)$
- $\lambda^e_{v_x}(r)$: probability of failing to report one or more targets when moving from $v_x$ towards $v_y$ with $r$ robots
- generalizes the label based algorithm
- complexity: $O(n \cdot B \cdot d^2 \cdot \log(d))$

A. Kolling, S. Carpin. *Probabilistic Graph Clear*, ICRA 2009
Answering probabilistic performance questions

- Given a desired \( p(T = 0|\mathcal{N}) \in [0, 1] \) how many robots are needed? For every vertex consider adjacent \( \lambda^e_{v_x}(r) \) and determine values exceeding the bound.

- Given \( r \) robots what is the strategy producing the highest \( p(T = 0|\mathcal{N}) \)?
Blocking and clearing at the same time
Line clear

- block and clear at the same time
- redefine weight function $w : (V \times E) \cup E \rightarrow \mathbb{N}$ and then compute labels


Clearing an environment with sweeping lines
On-line map discovery

- if the environment is not known upfront, it can be progressively discovered
- can be implemented with simple line following routines
- progressively builds a graph-clear instance
- not guaranteed to be optimal
- full paper presented Wednesday afternoon

On-line map discovery
Conclusions & Open questions

- Graph Clear is parametric with regard to blocking and clearing operations
- rich theoretical connections to graph theory
- can we determine provably good approximated solutions for graphs?
- how will be problem change if we minimize a different parameter?
- can we implement it on robots with minimal sensing capabilities?