A Visibility-Based Pursuit-Evasion Problem

Leonidas J. Guibas, Jean-Claude Latombe,
Steven M. LaValle, David Lin
et Rajeev Motwani

18/12/2K

Élodie-Jane Sims.

Le problème

- Capturer un évadé imprévisible dans un environnement polygonal
  - Pour un ensemble de poursuivants, planifier une stratégie de mouvement gagnante
  - Trouver le nombre minimal de poursuivants.

Plan de l'article

❶ Formulation mathématique du problème

❷ Bornes de complexité du problème

❸ Algorithme complet de calcul d'une stratégie gagnante

❹ Exemples implémentés par cet algorithme.

1. Définition du problème

Les poursuivants et l'évadé sont modélisés par des points qui se déplacent dans un espace libre polygonal, F.

- $e(t) \in F$: la position de l'*évadé* au temps $t \geq 0$
  - $e : [0, \infty) \rightarrow F$ fonction continue
  - l'évadé est capable de se déplacer à une vitesse arbitrairement grande
  - la position initiale $e(0)$ et le chemin e sont considérés comme inconnus des poursuivants
- $\gamma^i(t)$: la position du $i^{ème} poursuivant$ au temps $t \geq 0$
  - $\gamma^i : [0, \infty) \rightarrow F$ chemin continu du $i^{ème}$ poursuivant

Vocabulaire

- région **contaminée** : région de F qui peut contenir l'évadé
- région **nettoyée** : région non contaminée
- région **recontaminée** : région contaminée, nettoyée, à nouveau contaminée
- **stratégie (de mouvement)**, $\gamma = \{\gamma^1, ..., \gamma^N\}$: spécification des chemins continus pour tous les poursuivants
- $\boldsymbol{V(q)}$, $q \in F$: l'ensemble des points de F qui sont visibles depuis q
- **stratégie gagnante** : $\forall e : [0, \infty) \to F$ $\exists t \geqslant 0$, $\exists i \in \{1, ..., N\}$ tq $e(t) \in V(\gamma^i(t))$

- $\boldsymbol{H(F)}$: nombre minimum de poursuivants pour lequel il existe une stratégie gagnante pour F

2. Complexité du problème

– $\boldsymbol{n}$ le nombre de côtés de F

– $\boldsymbol{h}$ le nombre de trous de F (pour F multi-connexe)

① $H(F) = O(\lg n)$, cas s. connexe

② $H(F) = \Omega(\lg n)$, cas s. c.

③ $H(F) = O(h + \lg n)$

④ $H(F) = \Omega(\sqrt{h} + \lg n)$

⑤ pour $H(F) = 1$, la même région peut être recontaminée $\Omega(n)$ fois.

2.1 $H(F) = O(\lg n)$

Découpage en triangle par dichotomie, puis balayage.

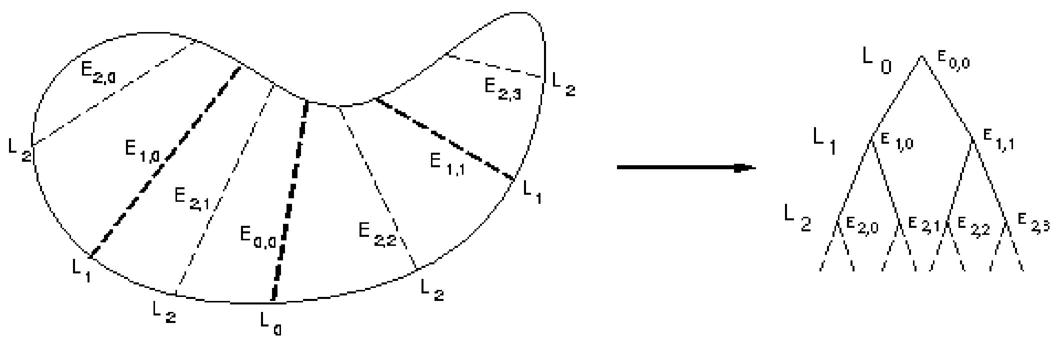


FIG. 1: *Partitionnement récursif pour déterminer une stratégie gagnante qui nécessite* $O(\lg n)$ *poursuivants*

2.2 $H(F) = \Omega(\lg n)$

Il suffit de trouver un exemple qui nécessite $\lg n$ poursuivants.

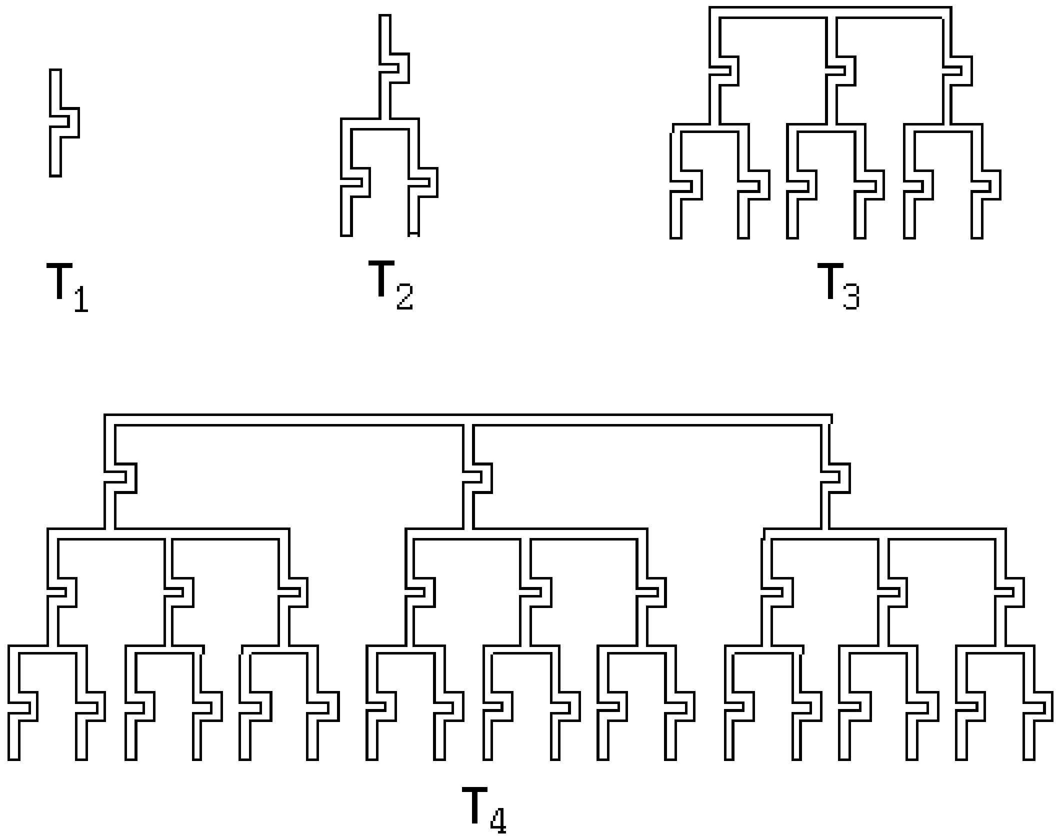


FIG. 2: *Construction systèmatique d'espaces libres simplements connectés qui nécessitent $\Omega(\lg n)$ poursuivants.*

2.3 $H(F) = O(h + \lg n)$

- Pb des trous : le poursuivant et l'évadé peuvent tourner autours sans se voir

  → au pire des cas on peut mettre un poursuivant entre chaque trou et le bord pour empêcher de tourner autour

2.4 $H(F) = \Omega(\sqrt{h} + \lg n)$

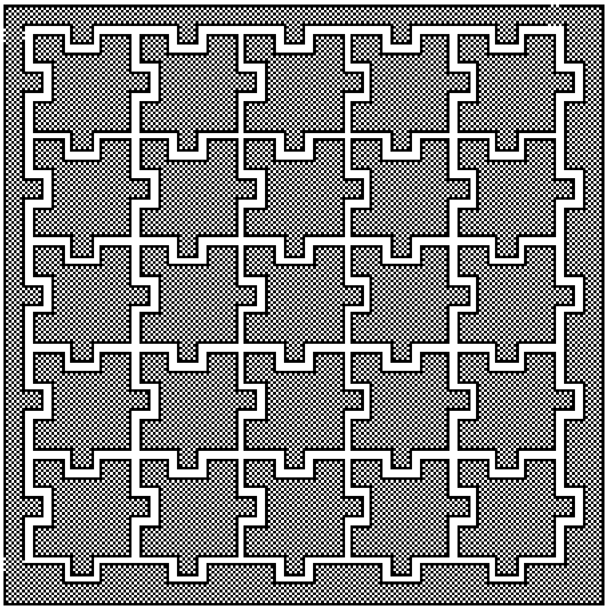


FIG. 3: *Un exemple qui nécessite un nombre de poursuivants qui est au moins proportionnel à la racine carrée du nombre de trous.*

2.5 $\Omega(n)$ recontaminations

Il existe des espaces libres simplement connexes nécessitant un seul poursuivant et $\Omega(n)$ recontaminations. (on ne connaît pas de borne supérieure).

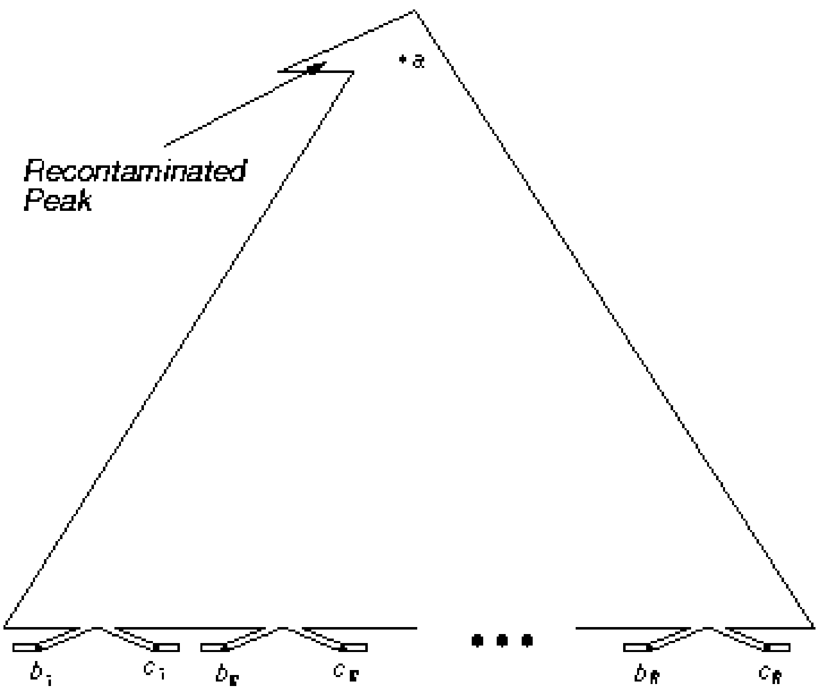


FIG. 4: *Un nombre linéaire de recontaminations est nécessaire. Pourtant ce polygone peut être cherché par un unique poursuivant, le pic doit être visité $k-1$ fois.*

3. Algorithme complet de calcul d'une stratégie gagnante

① Espace d'information

② Changements d'information critiques

③ Algorithme pour le cas d'un seul poursuivant

④ Plusieurs poursuivants.

3.1 Espace d'information

? position de l'évadé inconnue, on veut identifier les situations qui peuvent apparaître lors de l'exécution d'une stratégie de mouvement.

- **espace d'état** : X, coordonnées $x = (x^1, ..., x^N, x^e)$
  - x^i, $i \in \{1, ..., N\}$: la position du $i^{ème}$ poursuivant
  - x^e : la position de l'évadé
- $\boldsymbol{X^p} \subseteq X$: $x^p = (x^1, ..., x^N)$
- $\boldsymbol{S} \subseteq F$: ensemble des points contaminés de F
- **état d'information** : $\eta = (x^p, S)$: $x^p \in X^p$ et $S \subseteq F$
- **espace d'information**, $\mathcal{I}$: ensemble des états d'information possibles
- **chemin dans l'espace d'information**, $\eta(t) = (\gamma^1, ..., \gamma^N, S(t))$

- $\Psi(\eta, \gamma, t_0, t_1)$: état d'information obtenu à partir de η en appliquant γ de t_0 à t_1

3.2 Changements d'information critiques

On partitionne l'espace d'informations en classes d'équivalences.

❖ $D \subseteq X^p$ **conservatif** :
Si
– $\forall \eta \in \mathcal{I}$ t.q. $x^p \in D$,
– $\forall \gamma : [t_0, t_1] \rightarrow D$ t.q. γ continue,
– $\gamma(t_0) = \gamma(t_1) = x^p$,
alors $\eta = \Psi(\eta, \gamma, t_0, t_1)$.

Si on se déplace dans un ensemble conservatif, par n'importe quel chemin, on a toujours les mêmes points contaminés.

➜ une partition cellulaire de X
➜ recherche dans le graphe des cellules conservatives.

3.3 Algorithme pour un poursuivant

? On s'intéresse essentielement au cas $H(F) = 1$ car calculer $H(F)$ est *NP-dur*, càd qu'il faudrait essayer avec un poursuivant puis incrémenter le nombre de poursuivants.

▼ Maintenir les états d'information

▼ Décomposition cellulaire

▼ Recherche dans l'espace d'information

Maintenir les états d'information

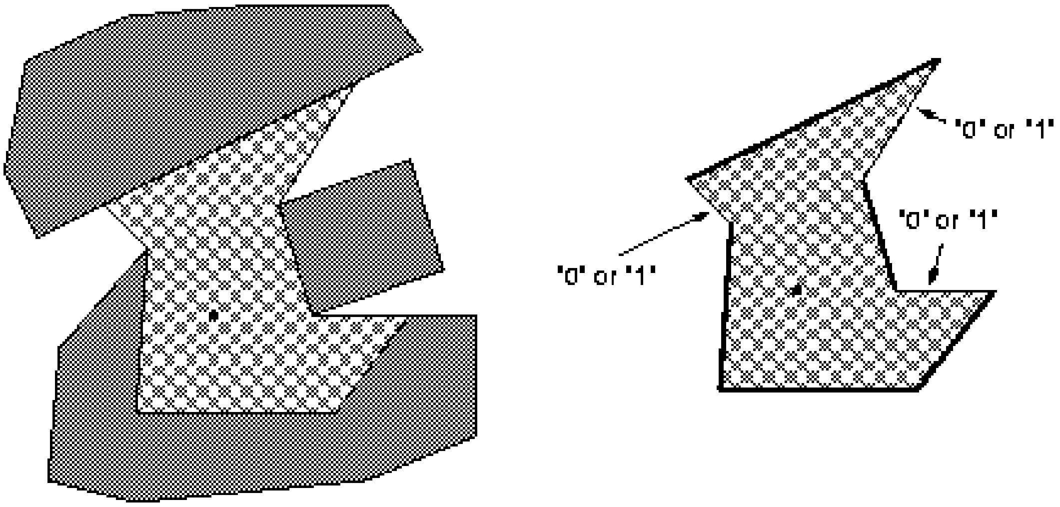


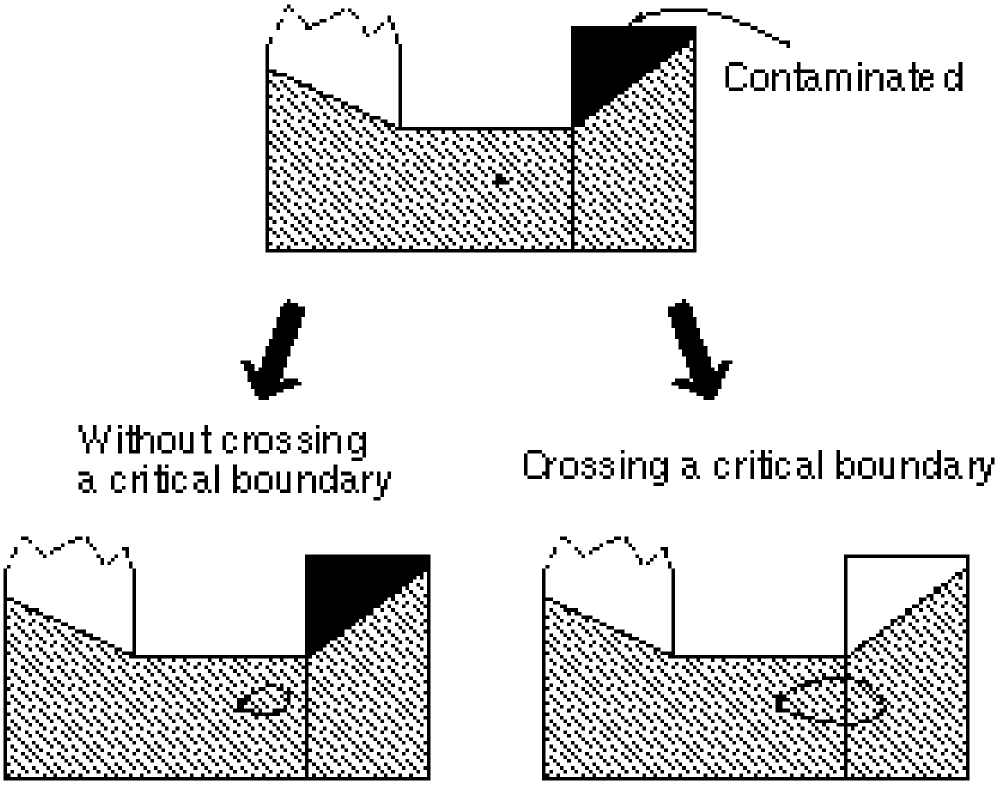
FIG. 5: *La labélisation des arrêtes peut être utilisée pour coder l'état d'information*

- Les cotés du polygone de visibilité non sur le bord de F sont labélisés :
  0 si l'espace libre contigu est nettoyé
  1 si l'espace libre contigu est contaminé (ou recontaminé).
- Les points contaminés sont bordés par :
  le bord de F
  des côtés du polygone de visibilité.

✥ **B(q)** : séquence binaire qui correspond aux labels assignés depuis $q \in F$

Décomposition cellulaire

Un évenement critique dans l'espace d'information ne peut apparaitre que quand les côtés de visibilité changent.



➜ une décomposition en cellules qui maintiennent les mêmes côtés labélisés contient seulement des cellules conservatives

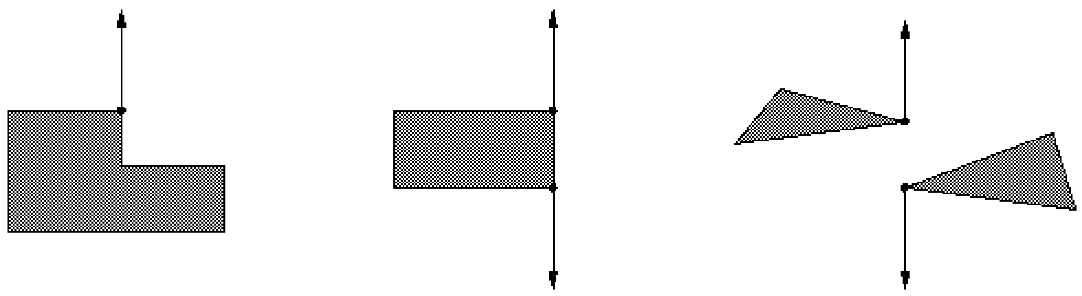


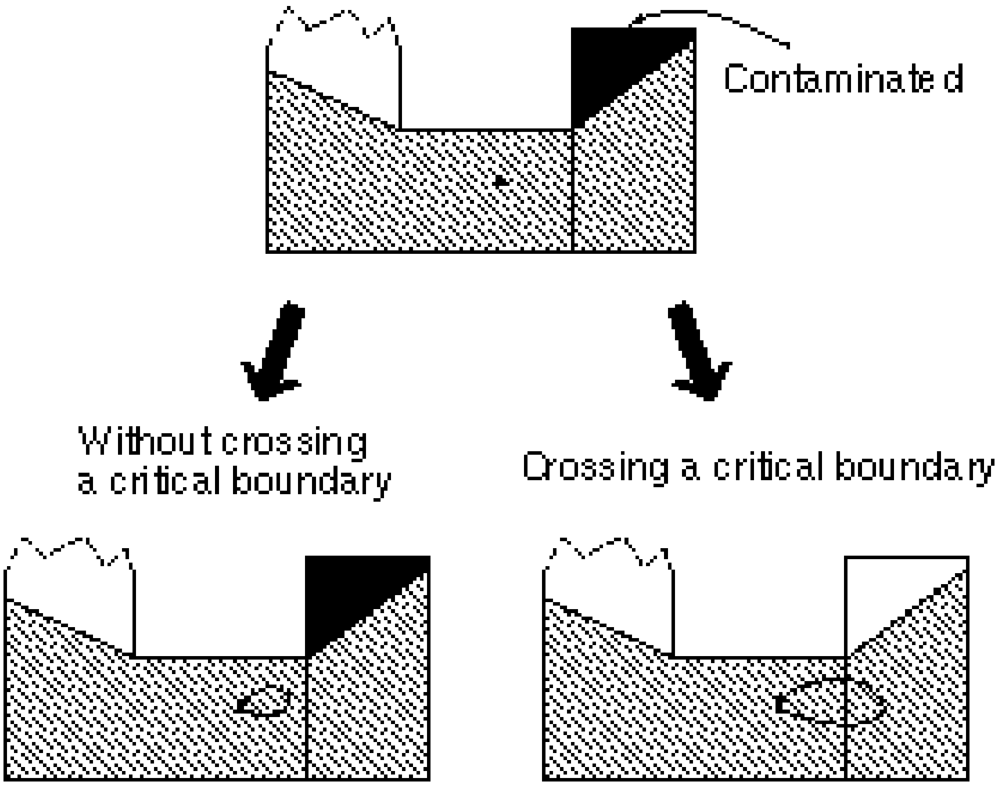
FIG. 6: *Le tir de rayons est exécuté pour trois cas généraux pour former les cellules d'arrêtes de visibilité.*

Recherche dans l'espace d'information

- Les cellules et leurs relations d'adjacence naturelles définissent un graphe, $\boldsymbol{G}_c$
- Pour chaque cellule on identifie un point $q \in F$
- départ : tous les labels sont à "1"
- but : tous les labels à "0"
- $\boldsymbol{G}_I$: graphe dirigé dérivé de G_c, pour lequel chaque sommet correspond à un sommet de G_c et à une instance de $B(q)$, (un sommet $= (q, B(q))$)

⇨ On utilise ensuite l'algorithme de Dijkstra dans G_I avec un coup qui correspond à la distance parcourue par le poursuivant dans l'espace libre.

Exemple



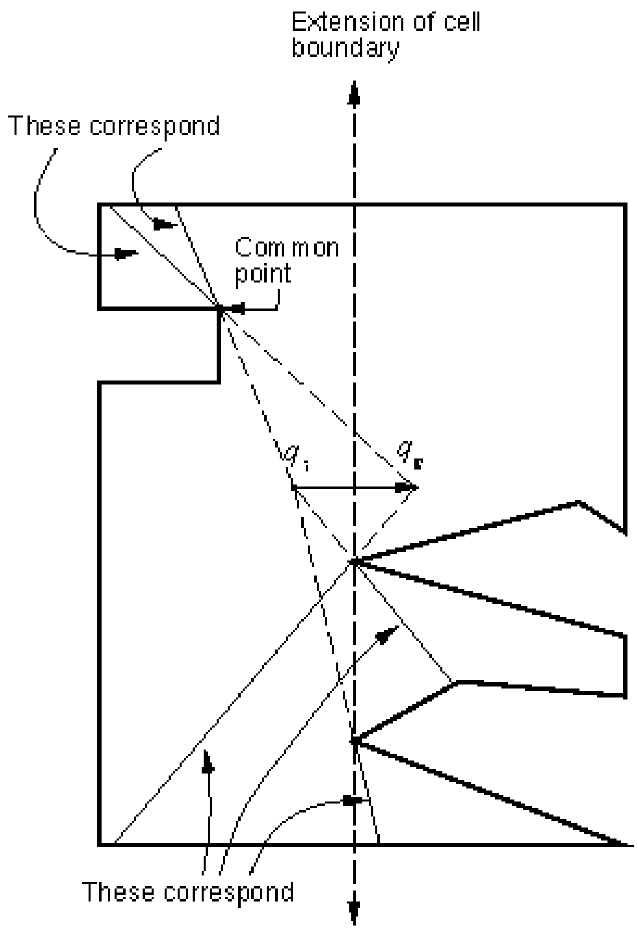
Soit le premier bit le label du côté labélisé le plus à gauche :

les arrêtes dirigées de G_I sont

– $(q_i, \text{“}01''\text{)}$ pointant sur $(q_j, \text{“}0'')$
– $(q_j, \text{“}0'')$ pointant sur $(q_i, \text{“}00'')$

Correspondances des côtés labélisés

Pour calculer les labels on établis des correspondances.



Fusions de côtes : si un des côté est labélisé “1”, le côté résultant est labélisé “1” (la contamination l'emporte).

3.4 Plusieurs poursuivants

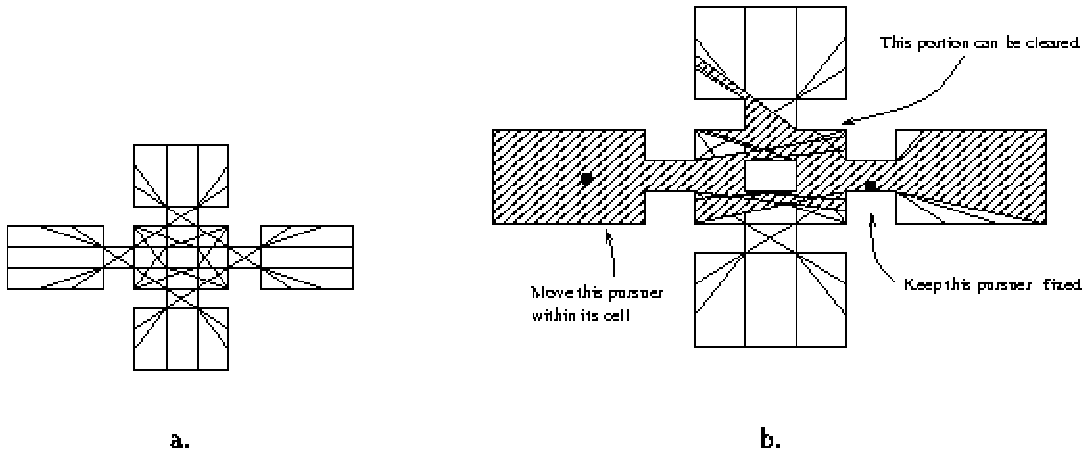


FIG. 7: *il est possible d'exécuter un chemin en boucle fermée qui change l'état d'information alors qu'il laisse les deux poursuivants dans leurs cellules de visibilité*

Le nombre de sommets dans le graphe de cellules est exponentiel sur le nombre de poursuivants,

→ pas de solutions efficaces actuellement

4. Exemples implémentés

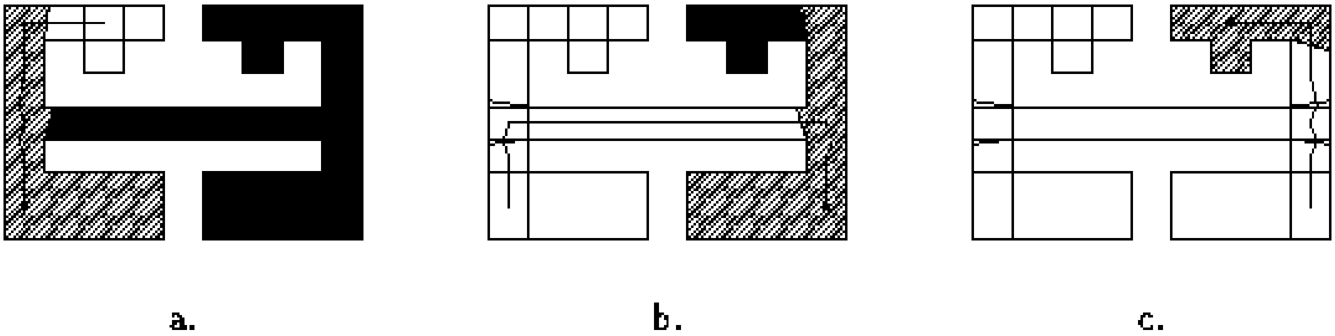


FIG. 8: *Une trajectoire gagnante calculée est montrée dans trois étapes. L'aire noire représente la région contaminée, l'aire blanche représente la région nettoyée. Le trait épais montre une portion de la trajectoire calculée, qui est continue dans chaque étape. La région hachurée indique la région de visibilité de la dernière position de la portion de trajectoire indiquée. Les trait fins dans la région nettoyée indique les bornes des cellules. Dans le dernier instantané, il n'y a plus d'endroit où l'évadé puisse être caché.*

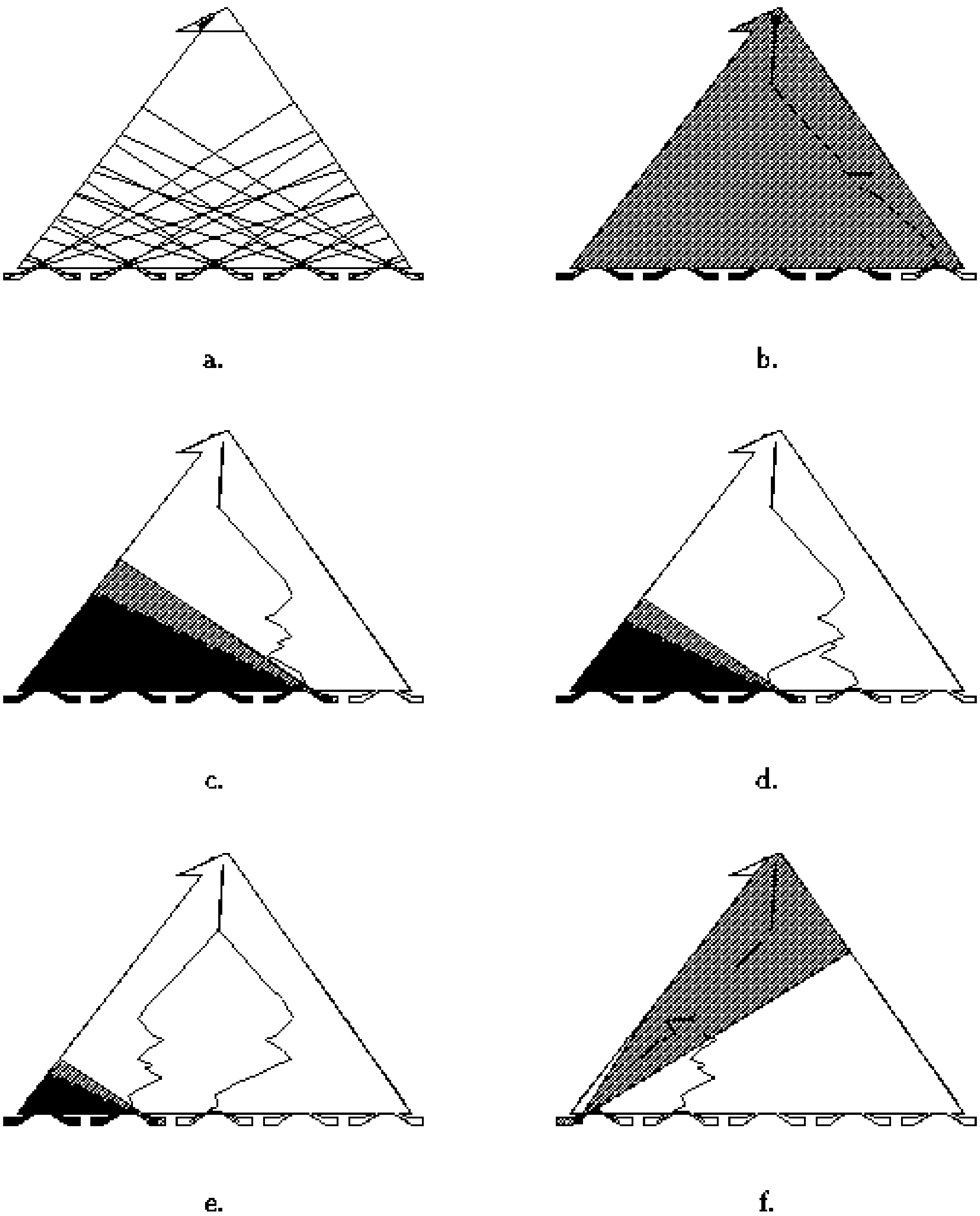


FIG. 9: *Cette exemple nécessite trois recontaminations, et fait parti des cas qui nécessitent un nombre linéaire de recontaminations.*

Problèmes liés

- ✗ Nettoyage d'un labyrinthe
- ✗ Recherche d'un évadé dans un graphe.

Applications

- ✈ Contrôle du trafic aérien
- ✯ Système de sécurité d'un immeuble avec une patrouille de robots mobiles
- ☛ Stratégie militaire
- ✚ Recherche de personnes lors de sauvetages

...

Exemple

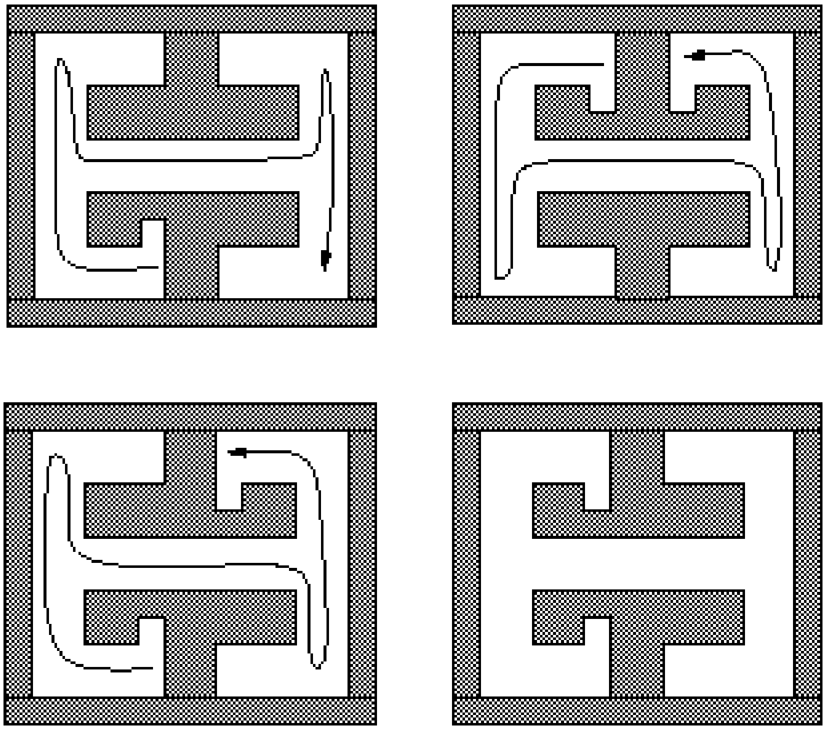


FIG. 10: *Quatre exemples qui ont des géométries similaires. L'exemple en bas à droite nécessite deux poursuivants, alors que les trois autres n'en nécessitent qu'un.*