Exposé d'un article pour l'examen de robotique

Élodie-Jane Sims.

6 août 2003

A Visibility-Based Pursuit-Evasion Problem

Leonidas J. Guibas, Jean-Claude Latombe, Steven M. LaValle, David Lin et Rajeev Motwani[1]

Proc. 5th Int. Workshop on Algorithms and Data Structures (WADS'97), Halifax, Nova Scotia, Canada; published as Lecture Notes in Computer Science 1272, F. Dehne, A. Rau-Chaplin, J.R. Sack, and R. Tamassia (eds.), pp. 17-30, 1997.

Résumé

Ce papier aborde le problème de la planification de mouvement pour un ou plusieurs poursuivants dans un environnement polygonal pour eventuellement "voir" un évadé qui est imprévisible, a une position initiale inconnue, et est capable de se déplacé arbitrairement vite. Une région de visibilité est associée à chaque poursuivant, et le but est de guarantir que l'évadé finira par être dans au moins dans une région de visibilité. L'étude de ce problème est motivée en partie par ses applications en robotique, telle la surveillance avec un robot mobile équipé d'une caméra qui doit trouvé une cible se déplaçant dans un espace de travail encombré.

Quelques bornes de complexité sont introduites, et un algorithme complet est présenté pour calculer une stratégie de mouvement gagnante. Pour un espace libre simplement-connecté, une borne logarithmique est établie sur le minimum de poursuivants nécessaires. Des bornes encadrantes pour des espaces libres "multiply-connected" sont aussi données. Un ensemble de problèmes qui sont résolvable par un seul poursuivant et qui demandent un nombre linéaire de recontaminqtion est présenté. "The complete algorithm searches a finite cell complex that is constructed on the basis of critical information changes." Ce concept peut être appliqué au principe des problèmes à plusieurs poursuivants, et le cas d'un unique poursuivant a été implémenté. Plusieurs stratégies gagnantes sont montrées, la plupart sont calculées en quelques secondes sur une station de travail standard.

1 Introduction

Le problème général abordé dans ce papier est une extension ou une combinaison de problèmes qui ont été considérés dans différents contextes. Des résultats intéressants ont été obtenus pour le problème de poursuite-évasion dans un graphe, dans lequel les poursuivants et l'évadé peuvent bouger d'un sommet à l'autre jusqu'à ce que éventuellement un poursuivant et l'évadé soient placés sur le même sommet. Le *nombre de recherche* d'un graphe fait référence au nombre minimal de poursuivants nécessaires pour résoudre le probléme de poursuite-évasion, est très lié à d'autres propriétés des graphes comme la "profondeur de coupe" (cutwidth). Les scénarios de poursuite-évasion dans des espaces continus ont amenés à une variété d'applications comme le contrôle du trafic aérien, la stratégie militaire, et la recherche de trajectoires. Ceci a conduit à une étude formelle des problèmes généraux de décisions dans lesquels deux décideurs ont des intérêts diamétralement opposés. Classiquement les jeux de poursuite-évasion exposent différents modèles de mouvement pour deux opposants, et des conditions de capture ou des stratégies optimales sont utilisées. "For exemple, in terms of the nonhololomic turning-radius constraints of the pursuer and evader". Bien que des problèmes de décisions intéressants proviennent des différents modèls de

1. *Computer Science Department, Stanford University*

déplacement, les contraintes géométriques de l'espace libre ne sont pas habituellement considérées dans les jeux classiques de poursuite-évasion. Une fois ces contraintes introduites, le problème hérite de complications provenants de planification de mouvement géométrique.

Une région de capture est souvent associée à un problème de poursuite-évasion, et la "capture" pour notre problème est définie par avoir l'évadé en ligne-de-vue d'un poursuivant. Un polygone de visibilité se déplaçant dans un environnement polygonal ajoute des informations géométriques qui doivent être utilisées, et aussi conduit à des connections avec le problème statique du musée. À la limite, les résultats du musée servent à une borne supérieure sur le nombre de poursuivants en couverant l'espace libre par des gardes statiques, guarantissant que tout évadé soit immédiatement visible. Après moins de gardes sont nécessaires si on les autorise à bouger et chercher l'évadé; cependant, la statégie de mouvement nécessaire peut devenir assez complexe. Un problème proche de celui du musée est celui de la tournée du gardient. Dans ce cas un chemin clos de longueure minimale est calculé tel que tout point du polygone est visible par un point le long du chemin. Dans notre cas, les poursuivants ont la charge supplémentaire d'assurer que l'évadé ne peut pas se "faufiler" dan une portion de l'environnement qui a déjà été exploré. Il a déjà été établi qu'il est difficile de déterminer si un polygone est cherchable par un unique poursuivant.

Plusieurs applications peuvent être envisagés pour des problèmes et des stratégies de mouvements de ce type. Par exemple, supposons un système de sécurité d'un immeuble utilisant quelques robots mobiles aves des caméras ou des capteurs qui peuvent détecter un intru. Un chemin pour la patrouille peut être automatiquement calculé guarantissant que tout intru mobile soit trouvé. Pour optimiser les dépenses, il serait aussi intéressant de connaître le nombre minimal de robots nécessaires. Les applications ne sont pas nécessairements limitées à des objectifs adverses. Par exemple, la tâche peut être de localiser automatiquement d'autres robots mobiles, objets dans un entrepôt ou une usine qui peuvent être déplacés pendant le processus, ou aussi des gens lors de recherche pour sauvetage. De telles stratégies peuvent être utilisé par des systèmes automatiques ou des chercheurs humains.

La Section 2 présente une formulation mathématique précise du problème. La Section 3 présente différentes bornes de complexité sur le nombre de poursuivants nécessaires et des problèmes liés. La Section 4 un concepte général pour réduire le problème à une recherche dans un graphe fini and un algorithme complet pour calculer une stratégie gagnante pour un espace libre donné. La Section 5 présente plusieurs exemples de stratégies gagnantes qui ont été calculées en utilisant l'implémentation de leur algorithme, and débat de quelques possibilités pratiques d'implémentations. Les conclusions sont présentées dans la Section 6.

2 Définition du problème

Les poursuivants et l'évadé sont modélisés par des points qui se déplacent dans un espace libre polygonal, F. Soit $e(t) \in F$ la position de l'*évadé* au temps $t \geq 0$. Nous supposons que $e : [0, \infty) \to F$ est une fonction continue, et 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. Toute région de F qui peut contenir l'évadé est appellée *contaminée*, sinon elle est appellée *nettoyée*. Si une région est contaminée, puis nettoyée, et à nouveau contaminée, elle sera appellée *recontaminée*.

Soit $\gamma^i(t)$ la position du $i^{ème}$*poursuivant* au temps $t \geq 0$. Soit γ^i le chemin continu du $i^{ème}$ poursuivant de la forme $\gamma^i : [0, \infty) \to F$. Soit γ une *stratégie (de mouvement)*, qui est la spécification des chemins continus pour tous les poursuivants: $\gamma = \{\gamma^1, ..., \gamma^N\}$.

Pour tout point, $q \in F$, soit $V(q)$ l'ensemble des points de F qui sont visibles de q (i.e., le segment linéaire joiniant q et tout point de $V(q)$ est dans F). Une stratégie, γ, est une *stratégie gagnante* si pour toute fonction continue $e : [0, \infty) \to F$ il existe un temps $t \in [0, \infty)$ et $i \in \{1, ..., N\}$ tels que $e(t) \in V(\gamma^i(t))$. Ceci implique que l'évadé sera finalement vu par un ou plusieurs poursuivants, quel que soit son chemin. Soit $H(F)$ le nombre minimum de poursuivants pour lequel il existe une stratégie gagnante pour F.

La Section 3 présente quelques bornes sur $H(F)$ pur des classes d'espace libre, et montre aussi que des polygones, pour lesquels $H(F) = 1$, n'admettent que des solutions qui demandent un nombre lineaire de recontaminations. La Section 4 considère le problème du calcul d'une stratégie gagnante, γ, pour un F donné.

3 Bornes majorantes

Plusieurs nouvelles bornes de complexité sont présentées dans cette section. Pour un espace libre simplement connecté, F, avec n cotés, il est montré que $H(F) = \Theta(\lg n)$. Pour un espace libre, F, avec h trous, il est montré que

$H(F) = \Omega(\sqrt{h} + \lg n)$ et $H(F) = O(h + \lg n)$. Pour la classe de problème pour lesquels $H(F) = 1$, il est montré que la même région peut être recontaminée $\Omega(n)$ fois. Ce résultat est surprenant car le problème de poursuite-évasion dans un graphe est connu pour ne demander aucune recontamination.

Consisdérons le problème de la détermination du nombre minimum de poursuivant, $H(F)$, nécessaires pour trouver un évadé dans un espace libre donné F. Ce nombre va généralement dépendre à la fois de la complexité topologique et géométrique de F. Une classe de simple polygones a été établie pour lesquels un unique poursuivant suffit (ressemblants à des "hérissions"). Pour tout F qui a au moins un trou, il est clair que au moins deux poursuivants sont nécessaires; si un unique poursuivant est utilisé, l'évadé peut toujours de déplacer autours du trou de telle manière que le trou soit toujours entre le poursuivant et l'évadé. Dans certains cas, des changements de géométrie minimes peuvent affecter significativement $H(F)$. Considérons par exemple, le problème de la Figure 1. Bien que les problèmes sont similaires, seul le problème en bas à droite nécessite deux poursuivants.

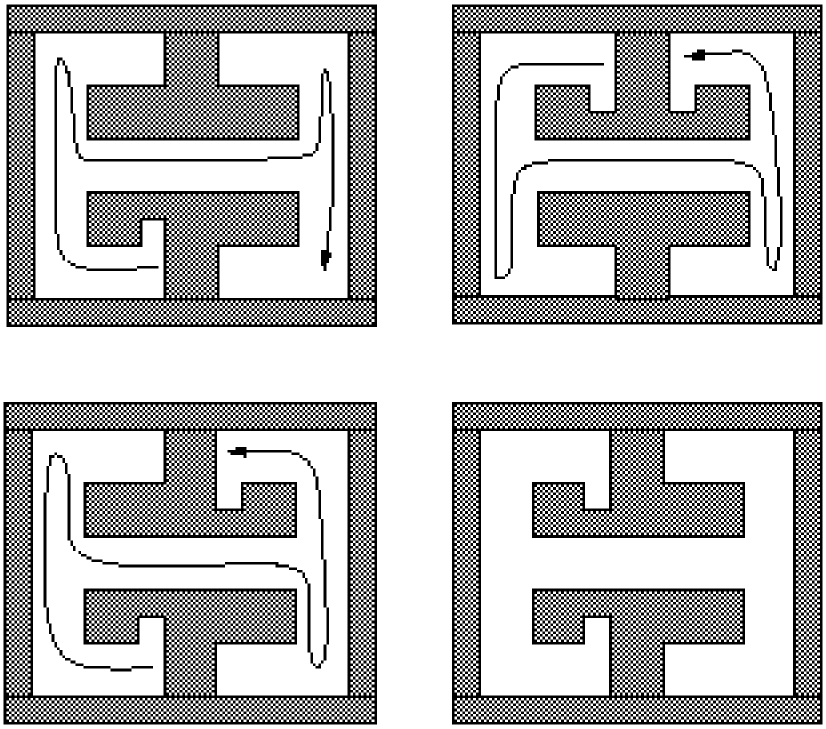

FIG. 1: *Quatre exemples qui ont des géométries similaires. L'exemple en bas à droite nécessite deux poursuivants, alors que les trois autres n'en nécessitent qu'un.*

Considérons $H(F)$ pour la classe des espaces libres simplement connectés. Soit n le nombre de côté de l'espace libre, qui est représenté par un polygone simple dans ce cas. Une borne logarithmique majorante est démontrée:

Théorème 1 *Pour tout espace libre simplement connecté F dans le pire des cas H(F) = O(lg n).*

Preuve: La preuve est basée sur un découpage en triangle du polygone par dichotomie (voir Fig. 2). On place un poursuivant par niveau de dichotomie (d'où $N = O(\lg n)$). Puis on deplace les poursuivant ..... A REDIRE □

La question restante pour les espaces libres simplement connectés est l'existence de problèmes qui demandent un nombre logarithmique de poursuivants. Quelques résultats de la recherche dans un graphe sont d'abord exposé et utilisés pour constuire des instances du problèmes du pire cas. Soit le *problème de Parsons* le problème de recherche dans un graphe. La tâche est de spécifier le nombre de poursuivants nécessaires pour trouver un evatdé qui execute un mouvement continu le long des arrêtes du graphe. Au lieu d'utiliser la visibilité, la capture à lieu auqnd un des poursuivants "touche" l'évadé. Soit G un graphe, et $S(N)$ le nombre de poursuivant nécessaires, appellé le nombre de recherche de G.

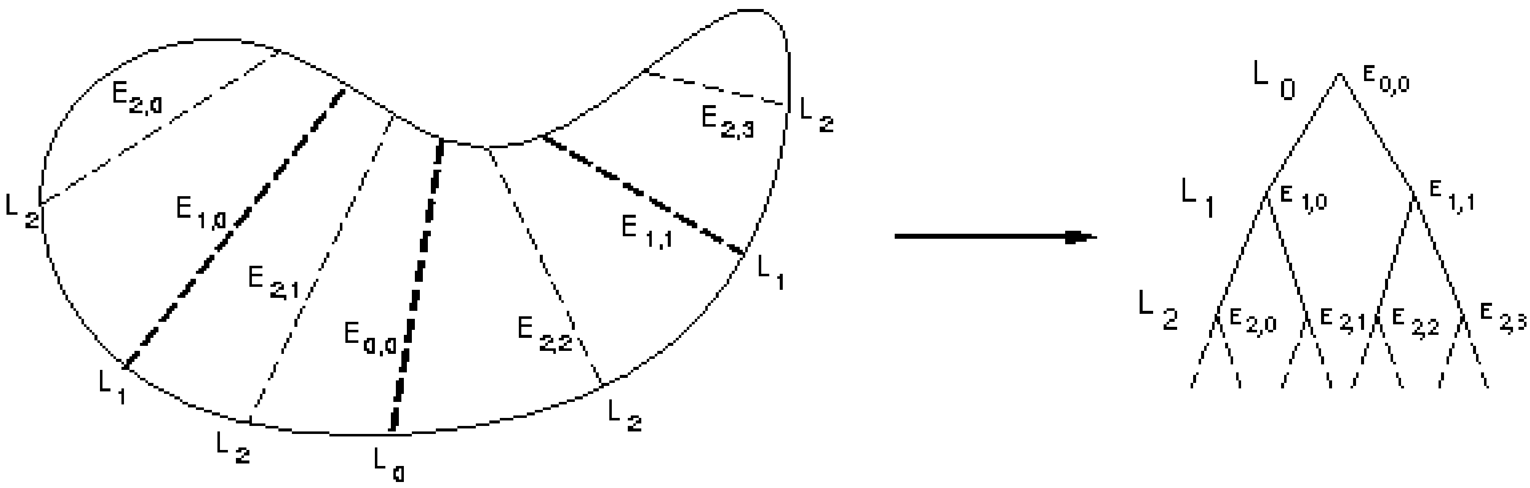

FIG. 2: *Partitionnement récursif pour déterminer une stratégie gagnante qui nécessite $O(\lg n)$ poursuivants*

Le lemme suivant implique qu'une réalisation géométrique de tout graphe planaire peut être construite:

Lemme 1 *Pour tout graphe planaire, G, il existe un espace libre polygonal F tel que le problème de Parsons sur G est équivalent au problème de pousuite-évasion basé sur la visibilité sur F.*

Preuve: La preuve prend comme equivalence une arrête de G et un corridor (voir Fig. 3). □

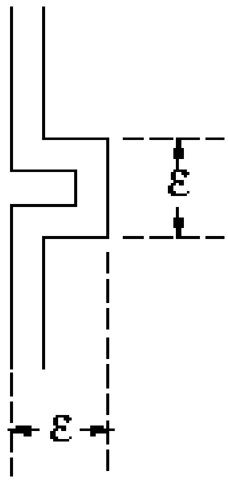

FIG. 3: *Un corridor de cette forme déconnecte au second-ordre la visibilité entre les deux entrées, et peut être utilisé pour construire une équivalence géométrique au problème de Parsons pour un graphe planaire.*

Le théorème construit une représentation géométrique du pire cas.

Théorème 2 *Il existe un espace libre simplement connecté F avec n cotés tel que $H(F) = \Omega(\lg n)$.*

Preuve: Voir Figure 4 □

Les Théorèmes 1 et 3 réunis impliquent une borne logarithmique pour la complexité exacte, $H(F) = \Theta(\lg n)$.

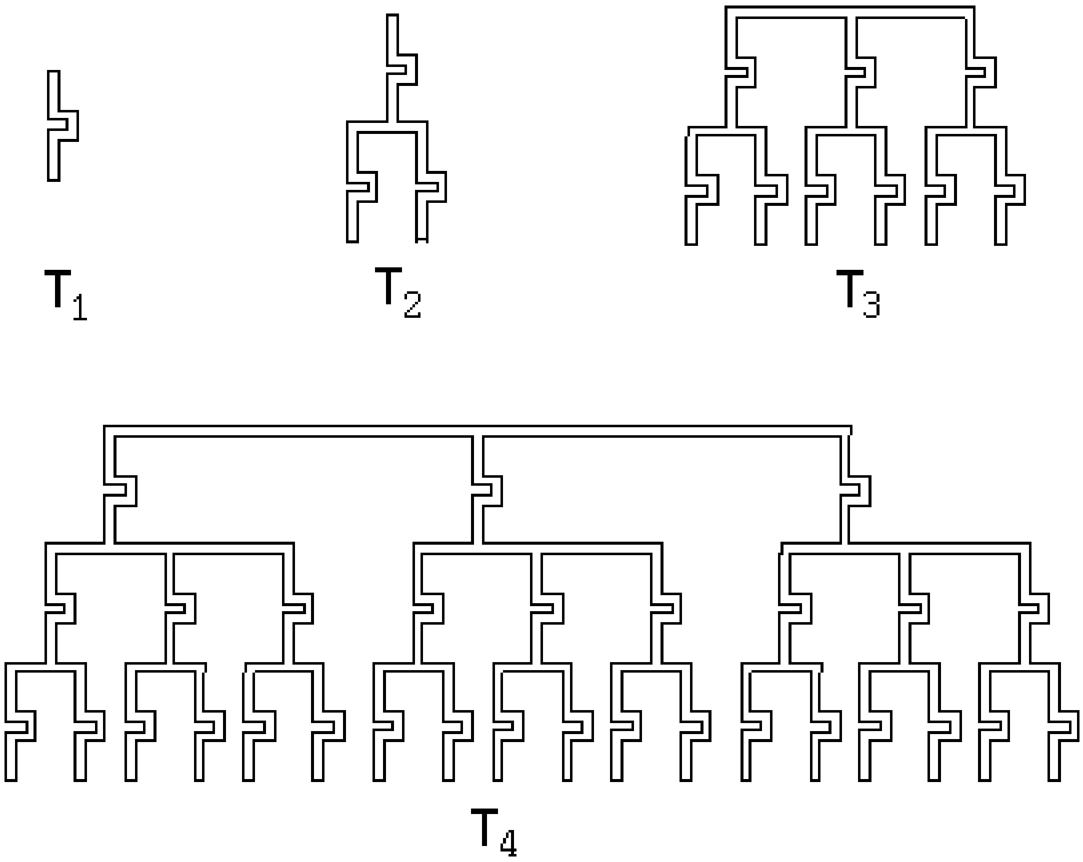

FIG. 4: *Construction systèmatique d'espaces libres simplements connectés qui nécessitent $\Omega(\lg n)$ poursuivants.*

Mainteant considérons la classe des problèmes pour lesquels F q h trous. Une borne inférieure en racine carrée et une borne supérieure linéaire sont obtenue, en fonction de h. Ceci evidement laisse le problème de déterminer une borne excte pour cette classe ouvert.

Théorème 3 *Pour tout espace libre F avec h trous dans le pire des cas $H(F) = O(h + \lg n)$.*

Preuve: L'idée de la preuve est de relier chaque trou à un côté du polygone extérieur de l'espace libre, on place un poursuivant sur chacun de ces traits ce qui empêchera l'évadé de tourner autours d'un trou, ce qui nous ramène au problème sans trous, d'où le résultat. □

Le Théorème 4 nous donne une borne inférieure.

Théorème 4 *Il existe un espace libre F avec h trous tel que $H(F) = \Omega(\sqrt{h} + \lg n)$.*

Preuve: La preuve consiste à prouver que c'est le cas pour la Figure 5 en s'appuyant sur la notion de profondeur de coupe d'un arbre. □

Le théorème final de cette section conserne la classe des espaces libres qui peuevent être cherchés par un unique poursuivant. Il montre qu'il existe des exemples qui nécessitent de recontaminer des portions de l'espace libre un nombre linéaire de fois. Ce résultat est surprenant car pour le problème de Parsons, il est montré qu'il n'y a aucune recontaminatin nécessaire. Un espace libre a déjà été donné qui nécessite deux recontaminations, qui au moins montre que la recontamination est généralement nécessaire pour le problème de poursuite-évasion basé sur la visibilité. Il apparaît trop souvent que ça peut être le pire des cas pathologiques; cependant, le Théorème 5 établit

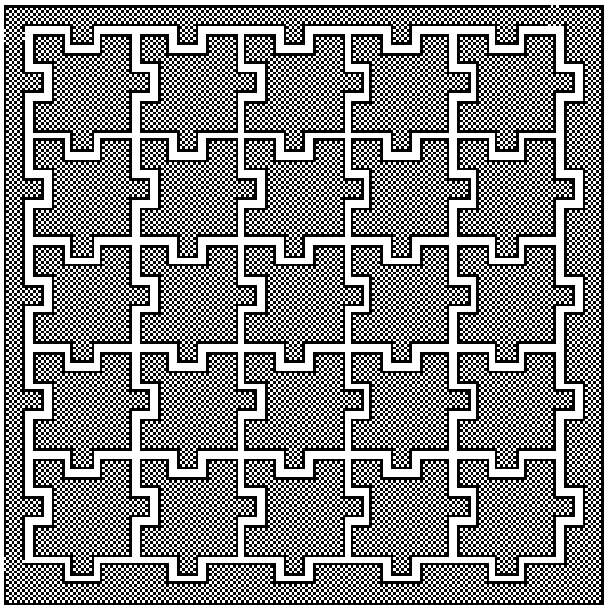

FIG. 5: *Un exemple qui nécessite un nombre de poursuivants qui est au moins proportionnel à la racine carrée du nombre de trous.*

qu'un nombre linéaire de recontaminatins peut être nécessaire, ce qui fait que le problème de poursuite-évasion basé sur la visibilité est significativement plus dur que le problème de graphe de Parsons. Le théorème donne une borne inférieure, et ça laisse le ouvert le problème de déterminer si le nombre de recontamination peut être borné supérieurement par un polynôme, ce qui impliquerait que le problème de décider si $H(F) = 1$ est NP-complet.

Théorème 5 *Il existe des espaces libres simplement connexes avec $H(F) = 1$ tels qu'ils nécessitent $\Omega(n)$ recontamination pour n côtés.*

Preuve: La preuve est basée sur l'exemple de la Figure 6. L'algorithme de nettoyage consiste a nettoyer dans l'ordre $(b_1, c_1, ..., b_k, c_k)$. On voit que si le "Recontamiated Peak", n'était pas là on réussirait à nettoyer le pseudo-triangle en balayant à partir du coin en bas à gauche (on voit ça en considérant les espaces visibles par le poursuivant). Si maintenant on ajoute ce "Recontamiated Peak", à chaque fois que le poursuivant sera en c_i, il perdra la vue de l'entrée au pic qui pourra donc être (re)contaminé par un évadé qui était caché dans un b_j ou c_j $(j > i)$, le poursuivant devra donc après chaque passage dans un c_i repasser par le point a pour décontaminer le pic. Le pic pourra donc être contaminé k fois, d'où le résultat. □

4 Calcul d'une stratégie gagnante

Alors que la Section 3 s'intéressait à borner la complexité pour certaines classes de problèmes, cette section traite de conceptes et d'algorithmes pour calculer une stratégie gagnante pour un espace libre donné. La Section 4.1 un espace d'information général (ou espace des états de connaissances) pour ce problème, et donne une méthode générale pour partitionner l'espace d'information en des classes d'équivalence, ce qui réduit le problème général à une recherche finie de cellules. La propriété de NP-dureté et aussi établie dans la Section 4.1. La Section 4.2 présente un algorithme complet pour le cas où $H(F) = 1$ qui calcule une stratégie gagnante en décomposant F en des cellules convexes basé sur la visibilité des côtés. Cette algorithme est assez efficace en pratique, et a été utilisé pour calculer les exemples montrés à la Section 5. Le cas où $H(F) > 1$ est traité à la Section 4.3.

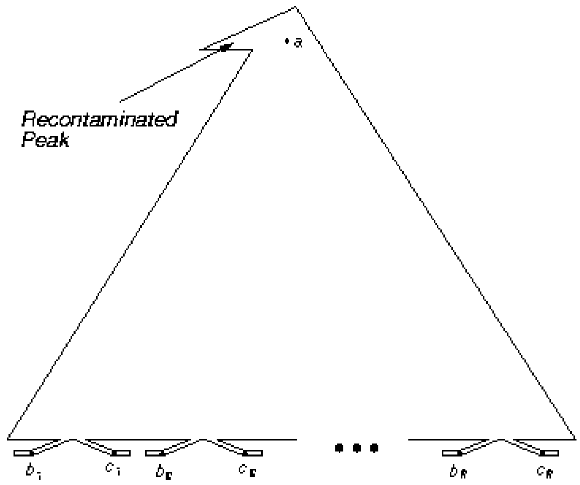

FIG. 6: *Un nombre linéaire de recontaminations est nécessaire. Pourtant ce polygone peut être cherché par un unique poursuivant, le pic doit être visité $k-1$ fois.*

4.1 Issues générales

En général, nous aimerions avoir un algorithme robuste et complet. Un algorithme *robuste* qui calcule une stratégie gagnante doit prendre F comme donnée et retourner γ qui garantirait que l'évadé soit forcément vu. Un algorithme *complet* doit calculer une solution gagnante pour un nombre donné de poursuivant, si elle existe. Un autre algorithme peut être utilisé pour calculer le nombre minimum de poursuivants nécessaire, ou l'algorithme complet doit être itéré en augmentant le nombre de poursuivants après chaque echec. Il est naturel de comparer la notion de complétude pour ce problème à la complétude pour le problème basique de planification de mouvement (i.e. l'algorthme trouvera un chemin sans collision, s'il existe). Une différence importante, cependant, est que le nombre *minimum* de poursuivants est crucial, mais n'a pas de correspondance pour le problème basique de la planification de mouvement. Un ensemble d'algorithmes heuristiques simples peuvent être dévellopés qui demandent plus de poursuivants que nécessaire (par exemple, trianguler l'espace de travail, et placer un poursuivant statique dans chaque triangle). Le problème devient plus difficile si on exige le nombre minimum de poursuivants.

Le problème général est intraitable si $P \neq NP$.

Théorème 6 *Calculer H(F) est NP-dur.*

Preuve: La preuve a déjà été faite pour le problème de Parsons, le Lemme 1 permet de conclure. □

Cela réduit significativement les espoirs qu'un algorithme efficace puissent exister pour le problème général; toutefois, un algorithme complet qui dépend exponentielement du nombre de poursuivant est présenté dans l'article (il est efficace en pratique pour $H(F) = 1$). Une direction de recherche future intéressante est de considérer un algorithme qui trouve des solutions utilisant un nombre de poursuivant "within q bound of optimal".

Un espace d'information Puisque la position de l'évadé est inconnue, nous n'avons pas un accés directe à l'état d'un moment donné. Cela induit à considérer un espace d'information qui identifie toute les situations uniques qui peuvent apparaître durant l'éxecution d'une stratégie de mouvement. Soit un *espace d'état*, X, représenté par les coordonnées $x = (x^1, ..., x^N, x^e)$, dans lequel x^i pour $1 \leq i \leq N$ représente la position du $i^{ème}$ poursuivant, et x^e représente la position de l'évadé. Puisque les positions des poursuivants sont toujours connues, soit X^p le sous-espace de X qui est représenté par les positions des poursuivants, $x^p = (x^1, ..., x^N)$.

Il est trés utile d'analyser la stratégie en terme de manipulation d'ensemble de positions possibles pour l'évadé. Soit $S \subseteq F$ l'ensemble de tous les points contaminés de F. Soit $\eta = (x^p, S)$, pour lequel $x^p \in X^p$ et $S \subseteq F$, l'*état d'information.* Soit l'*espace d'information*, $\mathcal{I}$, l'ensemble des états d'informations possibles. L'espace d'information est un outil de représentation standard pour des problèmes qui ont des états d'informations imparfaits, et a été très utile pour le contrôle optimal stochastique et la théorie des jeux dynamiques, et la planification de mouvement. Les états d'information correspondent aussi à l'état de connaissance "introduced for manipulation planning under uncertainty".

Pour un stratégie fixée, γ, un chemin dans l'espace d'information sera obtenu par $\eta(t) = (\gamma^1, ..., \gamma^N, S(t))$ dans lequel S(t) peut être déterminé par un $S(0)$ initial et les trajectoires $\{\gamma^i(t')|t' \in [0,t]\}$ pour tout $i \in 1, ..., N$. Soit $\Psi(\eta, \gamma, t_0, t_1)$ l'état d'information qui sera obtenu à partir de l'état d'information η, et en appliquant la stratégie γ de t_0 à t_1. La fonction Ψ peut être vue comme une "boîte noire" qui *produit l'état d'information résultant quand une portion d'une stratégie est exécutée.

Identification de changements d'information critiques Nous définissons ensuite un méchanisme général pour définir les changements d'information critiques. Ceci est inspiré en partie par une approche standard de la planification de mouvement, qui préserve la complétude en utilisant une décomposition de l'espace des configurations qui est construite en analysant les évènement critiques. Par exemple, une décomposition cellulaire peut être déterminée par l'analyse les "manifolds" de contact dans une configuration dans un espace des configurations composé qui est généré par les positions des plusieurs disques dans le plan.

La définition suivante décrit une propriété d'invariance d'information, qui autorise l'espace d'information, $\mathcal{I}$, à être partitionné en classes d'équivalences. Un ensemble connecté $D \subseteq X^p$ est *conservatif* si $\forall \eta \in \mathcal{I}$ tels que $x^p \in D$, et $\forall \gamma : [t_0, t_1] \to D$ telles que γ est continue et $\gamma(t_0) = \gamma(t_1) = x^p$, alors le même état d'information, $\eta = \Psi(\eta, \gamma, t_0, t_1)$, est obtenu. Ceci implique que l'état d'information ne peut pas changer en se déplaçant le long de chemins clos dans D. Comme dans le cas du déplacement dans un champ conservatif, ce qui implique:

Théorème 7 (Invariance de chemin) *Si D est conservatif alors pour toutes trajectoires conservatives γ_1, γ_2, dans D telles que $\gamma_1(t_0) = \gamma_2(t_0)$ et $\gamma_1(t_1) = \gamma_2(t_1)$ alors $\Psi(\eta, \gamma_1, t_0, t_1) = \Psi(\eta, \gamma_2, t_0, t_1)$, pour tout η.*

(i.e. dans un ensemble conservatif, si on va d'un point à un autre par n'importe quel chemin dans l'ensemble on arrive toujours avec mes mêmes points contaminés).
Preuve: La preuve consiste à prendre un troisième chemin γ_3, tel que $\gamma_3(t_0) = \gamma_1(t_1)$ et $\gamma_3(t_1) = \gamma_1(t_0)$, alors on a une boucle close en concaténant γ_3 et γ_2, d'après la définition d'ensemble conservatif on arrive au même état d'information donc on a bien $\Psi(\eta, \gamma_1, t_0, t_1) = \Psi(\eta, \gamma_2, t_0, t_1)$. □

Ainsi, l'état d'information en se déplaçant de $x_1^p\ inD$ à $x_1^p \in D$ est invariant du chemin choisi. Soit $\mathcal{D}$ un recouvrement de cellules conservatives qui forme une partition de X^p. La partition cellulaire, $\mathcal{D}$, de X peut être utilisée pour définir une relation d'équivalence, $\sim$, sur l'espace d'information, $\mathcal{I}$. Pour tous états d'information $\eta_1 = (x_1^p, S_1)$ et $\eta_2 = (x_2^p, S_2)$, nous avons $\eta_1 \sim \eta_2$ si il existe un $D \in \mathcal{D}$ avec $x_1^p, x_2^p \in D$, et si il existe γ dans D continue, t_0 et t_1 tels que $\gamma(t_0) = x_1^p, \gamma(t_1) = x_2^p$ et $\Psi(\eta_1, \gamma, t_0, t_1) = \eta_2$. En d'autres mots, deux états d'informations sont équivalents s'ils se projètent dans la même cellule, et l'une est atteignable depuis l'autre en executant un mouvment continu dans la même cellule. Soit $\mathcal{I}/\sim$ l'espace quotient dont les éléments sont les classes d'équivalences pour la relation d'équivalence $\sim$. Pour l'algorithmen complet, les éléments de $\mathcal{I}/\sim$ seront des sommets dans un graphe. Les arrêtes dans le graphe sont des paires de classes d'équivalences d'information qui peuvent êtres atteintes en se déplaçant d'une cellule conservative à une autre. Un tel graphe peut être cherché pour approximer une solution gagnante, sans s'intéresser au choix de chemins particuliers dans les cellules conservatives.

4.2 Un algorithme complet pour un poursuivant unique

Puisque le problème général est NP-dur, il vaut mieux se centrer sur l'algorithme complet pour le cas d'un poursuivant unique. L'idée basique et de particitionner l'espace libre en cellules convexes qui maintiennent la complétude, et exécuter une recherche sur l'espace d'information quotient qui en résulte. Cet algorithme a été implémenté et testé sur divers exemples, qui sont montrés en Section 5. Les solutions pour le cas de multiples poursuivants est reporté à la Section 4.3.

Maintenir les états d'information Supposons que le poursuivant est en un point $q \in F$. Considérons la séquence circulaire d'arrêtes dans la polygone de visibilité résultant. Les arrêtes généralement alternent entre un bord d'un obstacle et un bord dans l'espace libre. Voir Figure 7. Appelons toutes les arrêtes qui bordent l'espace libre *côtés troués*. Considérons l'association d'un label binaire à chaque côté troué. Si la portion de l'espace libre qui borde le côté troué est contaminée, alors on assigne un label "1"; sinon, on asigne un label "0" indiquant qu'il est propre.

Soit $B(q)$ la séquence binaire qui correspond aux label s assignés depuis $q \in F$. Notons que l'ensembles des points contaminés est borné par un polygone qui doit contenir doit des côtés de F soit des côtés troués du polugone de visibilité du poursuivant. Ainsi, la spécification de q et de $B(q)$ est uniquement caractérisée par l'état d'information.

Il y a une restriction supplémentaire sur $B(q)$ qui doit être renforcée. Si deux côtés troués sont sur le bord d'une même composante de $F \backslash V(q)$, alors ils doivent tout le temps avoir le même label. Si ce n'est pas le cas, alors la même composante sera identifiée comme étant à la fois recontaminée et nettoyée. En effectuant des analyses de connectivité, on peut reduire de façon conséquente le nombre de bits de $B(q)$; l'algorithme décrit dans cet section fonctionnera correctement dans les deux cas.

Une décomposition cellulaire qui préserve la complétude Considérons la représentation de l'état d'information utilisant q et $B(q)$, et considérons que le poursuivant se déplace sur un chemin continu en boucle close qui ne fait ni apparaître ni disparaître de côtés troués à aucun moment. Chaque côté troué va changer continuellement pendant le mouvement; cependant le label correspondant au côté troué ne va pas changer. L'état d'information ne peut pas changer sans qu'un côté troué n'apparaisse ou ne disparaisse. Par exemple, considérons le problème montré dans la Figure 8 qui montre un unique poursuivant qui s'approche de la fin d'un corridor. Si le mouvement en boucle close de la gauche est exécuté, la fin du corridor reste contaminée. Ceci implique que bien que l'état d'information change pendant le mouvement, l'état d'information d'origine est obtenu au retour. Pendant le mouvement en boucle close de la droite, le côté troué disparaît et réapparaît. Dans ce cas, l'état d'information à l'arrivée et différent. Le label du côté troué passe de "1" à "0".

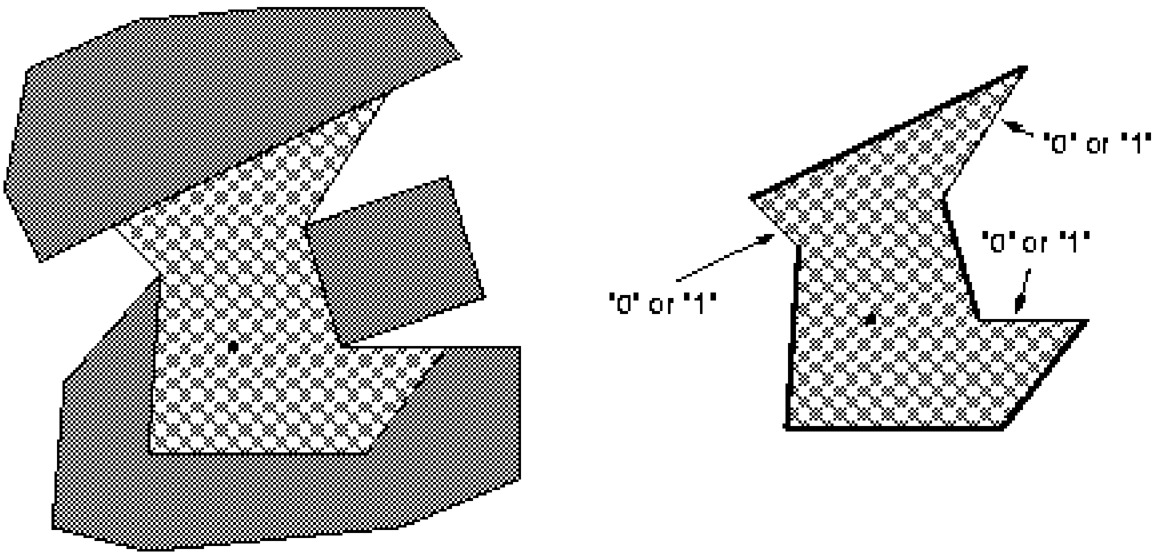

FIG. 7: *La labélisation des arrêtes peut être utilisée pour coder l'état d'information*

À partir de là, une décomposition en cellules qui maintient les mêmes côtés troués contiendra seulement des cellules conservatives. L'idée est de partitionner l'espace libre en cellules convexes en identifiant les endroits critiques pour lesquels les côtés de visibilité changent. Une décomposition de ce type a été utilisée pour la localisation de robot, et génère $O(n^3)$ cellules dans le pire cas pour un polygone simple (ce qui est toujours vrai si $H(F) = 1$). L'espace libre peut être suffisament partitionné dans notre cas en étendants les rayons des trois cas généraux montré à la Figure 9. Les côtés obstacles sont étendus dans l'une des directions, ou dans les deux si possible. Les paires de sommets sont étendues à l'extérieur seulement si les deux directions sont libres le long de la ligne dessinée entre la paire de points. Ceci empêche le cas pour lequel une direction ne peut pas être étendue; aussi les côtés de visibilité changent pour ce cas, ça représente un changement critique d'information. La Figure 16.a un exemple calculé pour cette décomposition en cellules.

Recherche de l'espace d'information L'étape suivante est la recherche de l'espace d'information pour une solution, ce qui correspond à spécifier l'ensemble des cellules adjacentes. La stratégie gagnante doit prendre la

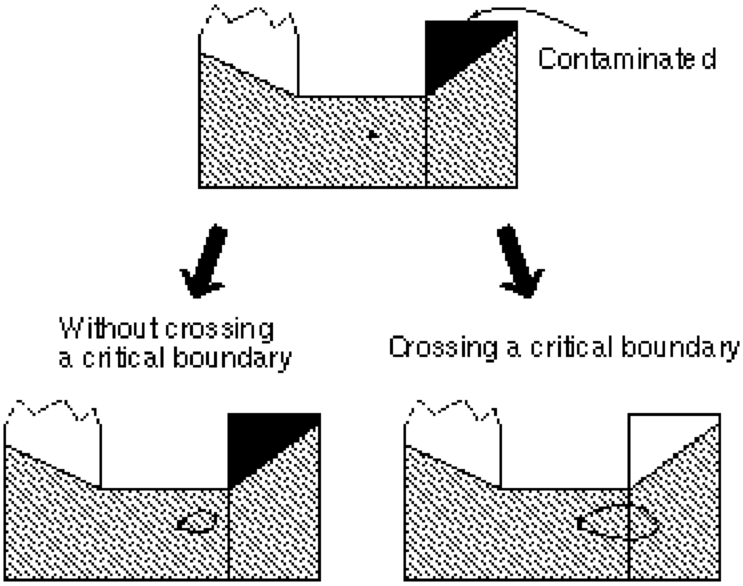

FIG. 8: *Un évenement critique dans l'espace d'information ne peut qu'apparaitre quand les côtés de visibilité changent.*

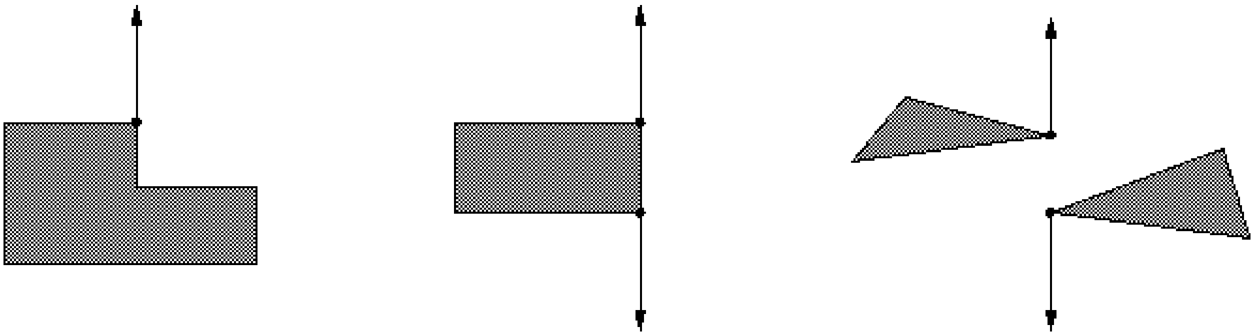

FIG. 9: *Le tir de rayons est exécuté pour trois cas généraux pour former les cellules d'arrêtes de visibilité.*

forme d'un chemin dans F. Ceci peut être construit en concaténant des segments de chemins linéaires, pour lesquels chaque segment connecte les centres d'une paire de cellules consécutives dans l'ensemble.

Les cellules et leurs relations d'adjacence naturelles définissent un graphe planaire fini, G_c appelé le *graphe cellulaire*. Les sommets dans G_c sont généralement visités plusieurs fois dans une sequence solution à cause des changements des états d'informations. Pour chaque sommet dans G_c, un point ,$q \in F$, dans la cellule correspondante est identifié, et les labels $B(q)$ peuvent êtres différents à chaque visite. Initialement, le poursuivant sera à une position pour laquelle tous les labels de trous sont "1". Le but est de trouver toute sequence de cellules dans G_c qui laisse le poursuivant à une position pour laquelle tous les labels de trous sont "0".

Un *grpahe d'états d'information* dirigé, G_I, peut être dérivé de G_c, pour lequel chaque sommet est visité au moins une fois durant l'éxecution d'une stratégie gagnante. Pour chaque sommet dans G_c, une ensemble de sommets sont inclus dan sG_I pour chaque $B(q)$ possible. Par exemple, supposons un sommet dans G_c représentant quelques cellules dans D, et supposons qu'il existe deux cotés troués pour $B(q)$ et pour tout $q \in D$. Quatre sommets seront inclus dans G_I qui tous correspondent au poursuivant dans la cellule de D; cependant, chaque sommet représente une unique possibilité pour $B(q)$: "00", "01", "10", ou "11". Identifions un sommet dans G_I par la paire $(q, B(q))$.

Pour compléter la construction de G_I, l'ensemble des arrêtes doit être défini. Ceci nécessite de déterminer les côtes troués approprié lorsque le poursuivant change de cellules. Supposons que le poursuivant se déplace de $q_i \in D_i$ à $q_j \in D_j$. Pour le cas simple montré en bas à droite de la Figure 8, supposons que le côté troué à gauche a au départ un label "0" et que le côté troué à droite a le label "1". Soit le premier bit le label du côté troué le plus à gauche. La première transition est de "01" à "0", et la seconde transition est de "0" à "00". Les arrêtes dirigées de G_I sont $(q_i, \text{"01"})$ pointant sur $(q_j, \text{"0"})$, $(q_j, \text{"0"})$ pointant sur $(q_i, \text{"00"})$.

Dans le cas de plusieurs côtés troués, les correspondances doivent être déterminées pour calculer correctement les labels des trous. Considérons l'exemple montré dans la Figure 10 qui illustre le cas général qui peut apparaître. Un côté troué de $V(q_i)$ correspond à un côté troué de $V(q_j)$ si ils partagent un sommet, et si ni l'un ni l'autre ne touche le prolongement de leurs bornes de cellules communes. Ce cas est montré en haut à droite de la Figure 10. Dans ce cas le label binaire sera préservé en se déplaçant directement de q_1 à q_2. Le cas est plus intéressant si les côtés troués touchent le prolongement de la cellule bornante, comme dans le bas de la Figure 10. En général, tous les côtés qui touchent l'extension en-dessous de la cellule correspondent les uns aux autres, et tous ceux qui touchent l'extension au-dessus de la cellule séparement les uns aux autres. Les transitions de ce types produisent essentielement la fusion ou la division des côtés troués. Il y a deux côtés troués dans la partie du bas de la Figure 10 quand le poursuivant est en q_1; alors qu'il n'y en a que un quand le poursuivant est en q_2. Lors d'une transition de q_1 à q_2, si les côtés troués de q_1 sont labelés "0" et "0", alors le côté troué correspondant de q_2 sera labelé "0". Si l'un des côtés troués de q_1 est labelé "1", alors le côté troué correspondant de q_2 sera labelé "1" (la contamination s'etend plus facilement). En général, si tout n côtés troués sont fusionnés, le côté troué correspondant recevra un label "1" si l'un des côtés troués originaux à un label "1".

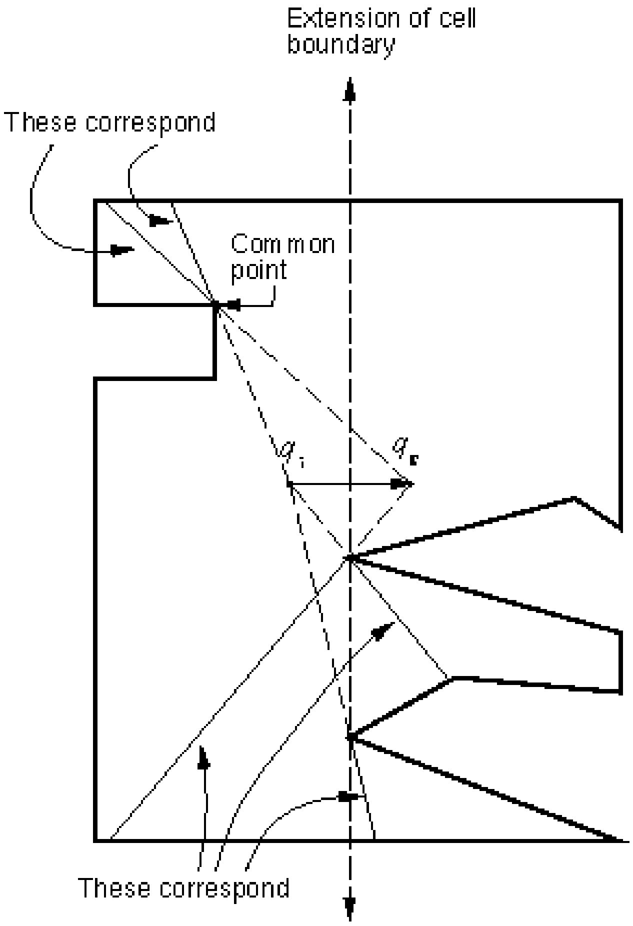

FIG. 10: *Les correspondances enre les côtés troués de différentes cellules voisines peuvent être directement déterminées. Les états d'information sont mis à jour lors des déplacements entre cellules en utilisant cette correspondance.*

Lorsque les correspondances de côtés troués sont déterminées, le graphe des états d'information peut être cherché en utilisant l'algorithme de Dijkstra avec un coup sur les arrêtesqui correspond à la distance parcourue dans l'espace libre par le poursuivant. Malheureusement, la complexité précise de l'algorithme complet ne peut pas être déterminée car c'est une question encore ouverte de savoir si le problème est en P ou NP. Dans le pire des cas, des exemples peuvent être construits qui sont exponentiel sur le nombre d'états d'information, mais il n'est pas clair si ces états d'information ont nécessairement besoin d'être représenté et cherché pour déterminer une solution (ou pour vérifier une solution).

4.3 Poursuivants multiples

En général, le concepte conservatif qui a été présenté en Section 4.1 peut être appliqué pour produire une décomposition cellulaire de X^p, qui est l'espace à $2N$-dimention qui code les positions des poursuivants. En raison de la complexité du problème, développer et implémenter un algorithme pratique est toujours un challenge même pour le cas de à deux poursuivants. Une décomposition cellulaire pour le cas à N poursuivants doit être construite de façon que les côtés dans l'union des N polygones de visibilité ne doit pas changer (les côtés ne doivent pas apparaitre ou disparaitre). Les cellules conservatives étaient formées en maintenant constant les côtes de visibilité pour le cas d'un poursuivant, et nous voudrions qu'un produit cartésien de cellules planaires puisse être formé pour directement définir des cellules conservatives pour plusieurs poursuivants; cependant, comme la Figure 11 l'indique, les cellules conservatives peuvent être considérablement plus compliquées pour le cas de deux poursuivants.Les côtés de visibilité des deux polygones de visibilité peuvent se coupés de telle façon que qu'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. Par exemple dans la Figure 11, supposons que le poursuivant à gauche se déplace légèrement vers la droite et retourne, pendent que le poursuivant de droite ne bouge pas. Ceci va nettoyer la portion indiquée sur la figure, même si les deux poursuivants restent dans leurs cellules. Le changement critique d'information apparait quand les somments des deux polygones de visibilité se rencontrent sur le côté d'un obstacle. Cette contrainte défini une algèbre à trois dimention dans $\Re^4$ "three-dimentional algebraic manifold in $\Re^4$" qui partitionne les cellules de X^p. Les contraintes algèbriques qui correspondent qui correspondent à ces types de cas augmentent significativement la difficulté de l'implémentation et ajoute de nombreuses cellules qui diminuent l'efficacité pratique.

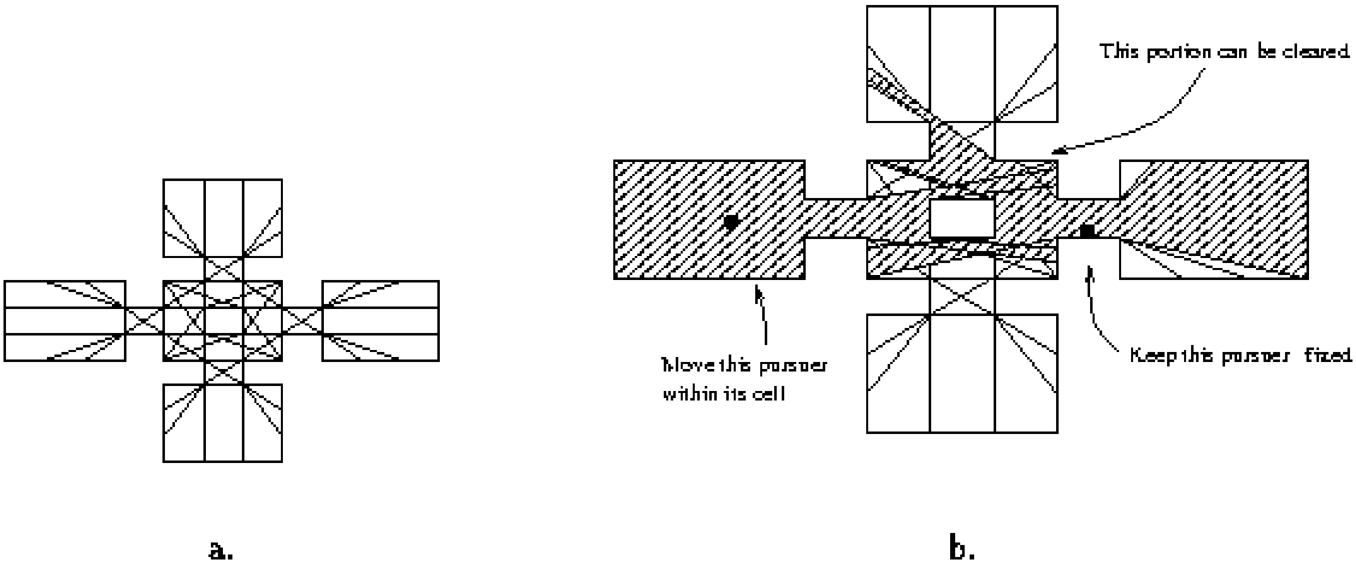

Fig. 11: *Cet exemple montre que l'état d'information peut être changé par un chemin en boucle fermé qui maintient constant les côtés de visibilité pour le cas de deux poursuivants: a) la décomposition en cellules basée sur les côtés de visibilité; b) le recouvrement de polygones de visibilité permet à l'état d'information de changer.*

Parce que le nombre de sommets dans le graphe des cellules est exponentiel sur le nombre de poursuivants, il reste de nombreuses voies de recherche d'efficacité à explorer. Beaucoup de cellules peuvent être combinées sans affecter la complétude s'il est montré que leurs limites communes ne représente pas un changement critique d'information. Le nombre de cellules peut être significativement réduit dans beaucoup de cas. Aussi, des stratégies de recherches habiles peuvent garder beaucoup de l'espace d'information en étant explorées. Une cause de cette réduction potentielle est le fait que de multiples solutions existent, incluant de multiples états qui chacuns peuvent représenter l'étape finale d'une stratégie gagnante. Même avec ces réductions, cependant, il semble peu problable qu'un algorithme pratique complet puisse être développé pour le cas où $H(F) \geq 3$.

5 Exemples implémentés

L'algorithme complet est implémenté en C++ et exécuter sur une station SGI Indigo2 avec un processeur MIPS R4400 à 200 Mhz. Les temps de compilations et d'autres paramètres pour plusieurs exemples sont listés dans la Figure 12. L'implémentation utilise utilise la structure de "quad-edge" pour maintenir l'ordre topologique sur les cellules conservatives. La stratégie de recherche est essentiellement l'algorithme de recherche du plus court chemin dans un graphe de Dijkstra, où les distances sont mesurées entre les différents centres des cellules adjacentes. La solution est calculée en traverant de centre de cellule à centre de cellule, ce qui amène dans la plupart des cas à ce que le chemin calculé pour le pousuivant soit dentelé. Dans quelques applications, il peut être approprié d'employer un algorithme de lissage pour le chemin pour respecter des contraintes additionnelles.

Problem	Edges	Nodes in G_c (Cells)	Nodes in G_I (Information)	Precomp. Time (sec)	Searching Time (sec)	Total Time
Fig. 13	28	25	200	0.04	0.02	0.06
Fig. 14	68	130	1727	0.44	0.12	0.56
Fig. 15	46	237	8787	0.53	1.59	2.12
Fig. 16	65	246	18830	0.87	9.86	10.73
Fig. 17	70	888	103049	3.00	168.63	171.63

FIG. 12: *Différentes statistiques sont montrés pour les exemples calculés.*

Les Figures 13-17 montrent différents exemples calculés. En raison du gsrand nombre de cellules conservatives, le Figures 15-17 sont illustrées avec la décomposition cellulaire dans des diagrammes séparés des diagrammes de solution. La Figure 15 montre l'"hookpin example". Notosn aue le crochet le plus à gauche et recontaminé deux fois. La Figure 16 est une instance de la séquence décrite dans la Section 3 qui nécessite un nombre linéaire de recontaminations. La région vers le haut de la figure est recontaminée 3 fois. Le dernier exemple génère un grand nombre de cellules conservatives, ce qui augmente significativement le temps de calcul.

6 Conclusions

Un problème de planification basé sur la visibilité a été identifié dans ce papier qui demande la recherche d'un évadé imprévisible dans un environnement polygonal. Cette tâche peut représenter une opération de base dans diverses application robotiques, telle la surveillance par des robots mobiles. D'autres applications potentielles peuvent être les opérations de recherche lors de sauvetages et la stratégie militaire.

Plusieurs limites sur la complexité ont été obtenues. Une borne logarithmique exacte sur le nombre de poursuivant nécessaires a été prouvé pour le cas d'espace libre simplement connecté. Un borne majorante linéaire et une borne minorante en racine carrée sur le nombre de trous ont été aussi démontrées. Il a aussi été montré qu'il existe des problèmes qui nécessitent un nombre linéaire de recontaminations. Il reste quelques problèmes ouverts, tel que la détermination d'une borne exacte du nombre de poursuivants pour un polygone général, et la détermination si une algorithme polynomial en temps esiste pour décider si $H(F) = 1$.

Le concepte d'espace d'information a été utilisé pour produire des caractéristiques naturelles des états du problème d'un unique poursuivant. Le problème de poursuite-évasion basé sur la visibilité a été montré comme NP-

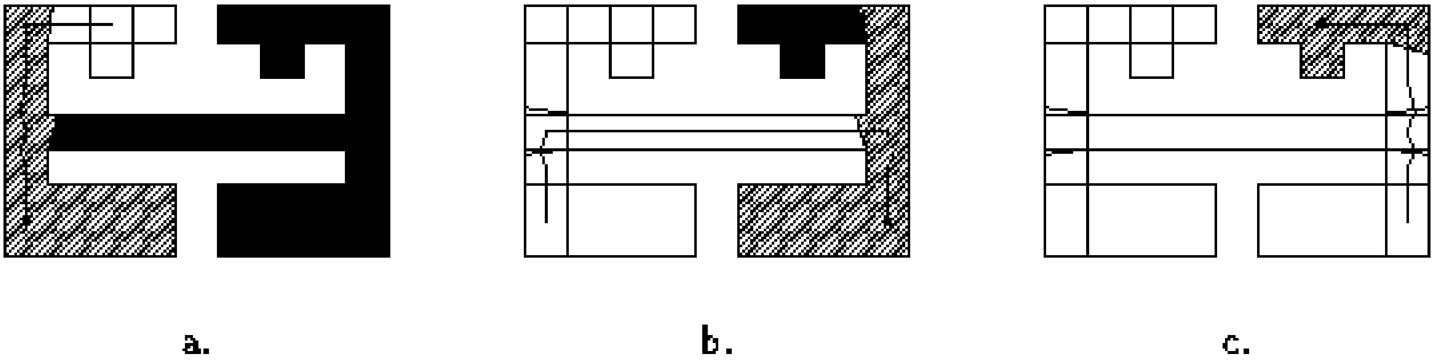

FIG. 13: *Une trajectoire gagnante calculée est montrée dans trois étapes. L'aire noire représente la région contaminée, l'aire blanche représente la région nettoyée. Le trait épais montre une portion de la trajectoire calculée, qui est continue dans chaque étape. La région hachurée indique la région 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.*

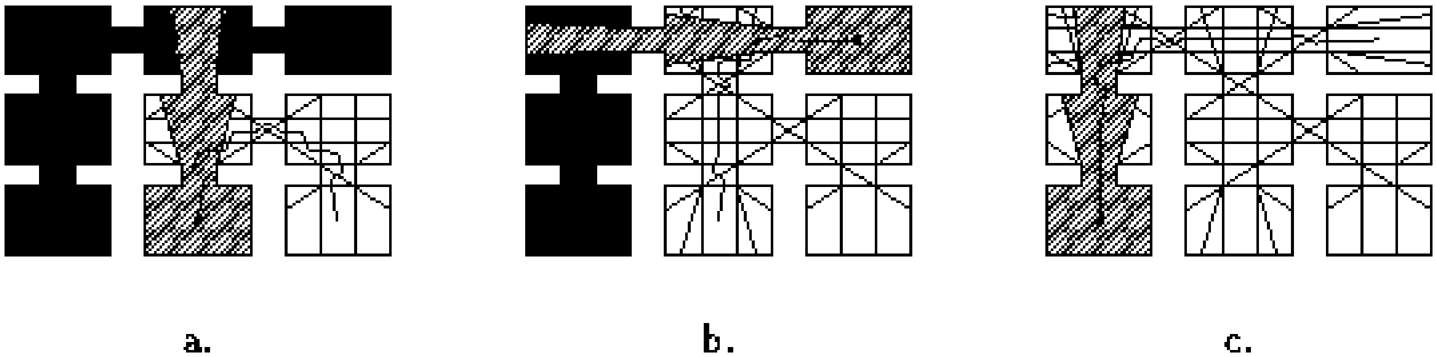

FIG. 14: *Un autre exemple calculé.*

dur. Le concepte général de partitionnement de l'espace d'information sur la base des changements d'information critiques a été introduit pour développer un algorithme complet. Pour le cas où $H(F) = 1$, l'algorithme complet a été implémenté, et plusieurs exemples ont été montré qui se calculent en quelques secondes ou moins sur une station de travail standard. Il reste de nombreuses possibilités d'implémentation pour le cas où $H(F) = 2$, en général, des algorithmes approximatifs peuvent produire les seuls espoirs d'obtenir en pratique des solutions à beaucoup de problèmes.

Plusieurs variations et extensions du problème sont pire à étudier. En ajoutant à une région de visibilité, chaque poursuivant peut avoir uen région de capture, at la tâche peut être de capturer l'évader en utilisant un ou plusieurs poursuivants. En utilisant le modèle d'évadé courant, seules des solutions connectives deviennent critiques pour déterminer une solution gagnante; cependant, le problème peut être rendu plus compliqué en étendant le modèle pour inclure des limites de vitesse, ou des prévisions stochastiques????. Les solutions topologiques peuvent devenir significativement plus complexes pour des espaces libres en 3-D. Les cellules conservatives et les concepte de côtés de visibilité peuvent être appliqués pour les cas en 3-D, mais il faut faire face à de nombreuses difficultés pour produire un algorithme efficace. Un autre variation du problème est de considérer un angle de vision limité, ou un ensemble de rayons de vision. Un angle de vue limité peut apparaitre de façon réaliste dans des applications, et le problème peut être étendu à la planification de stratégies qui balayent des angles de vue en plus des poursuivants bougeants. Finalement, une fonction de coût peut être définie en plus, conduisant des problèmes comme trouver un évadé en un minimum de temps.

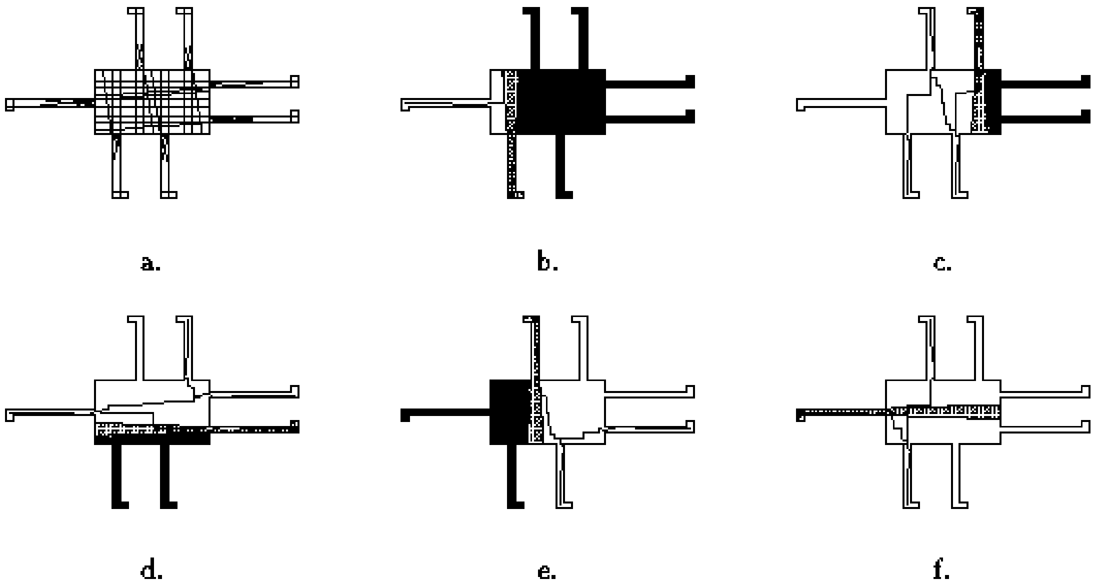

FIG. 15: *Cet exemple difficile nécessite deux recontaminations du corridor le plus à gauche.*

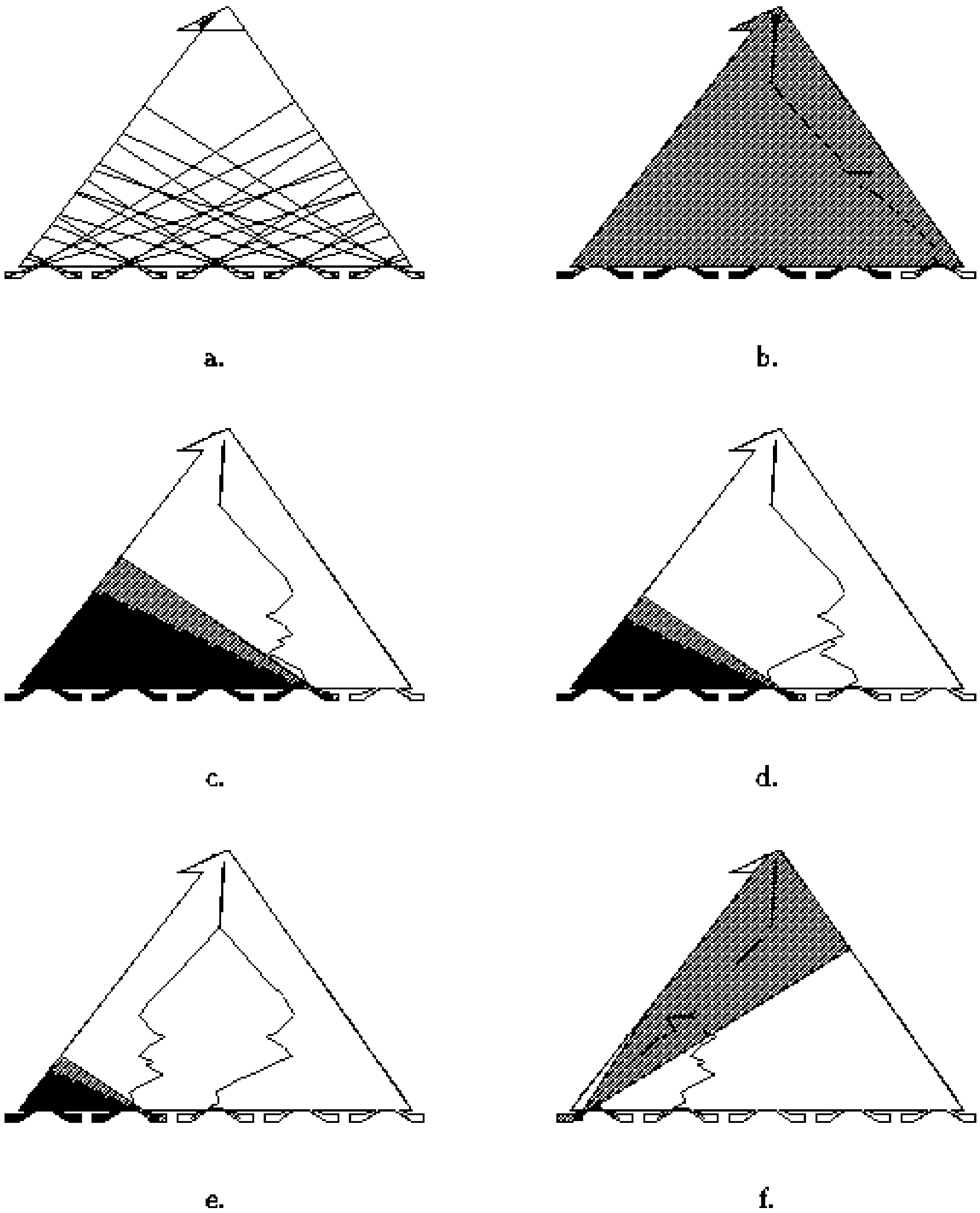

FIG. 16: *Cette exemple nécessite trois recontaminations, et fait parti des cas qui nécessitent un nombre linéaire de recontaminations.*

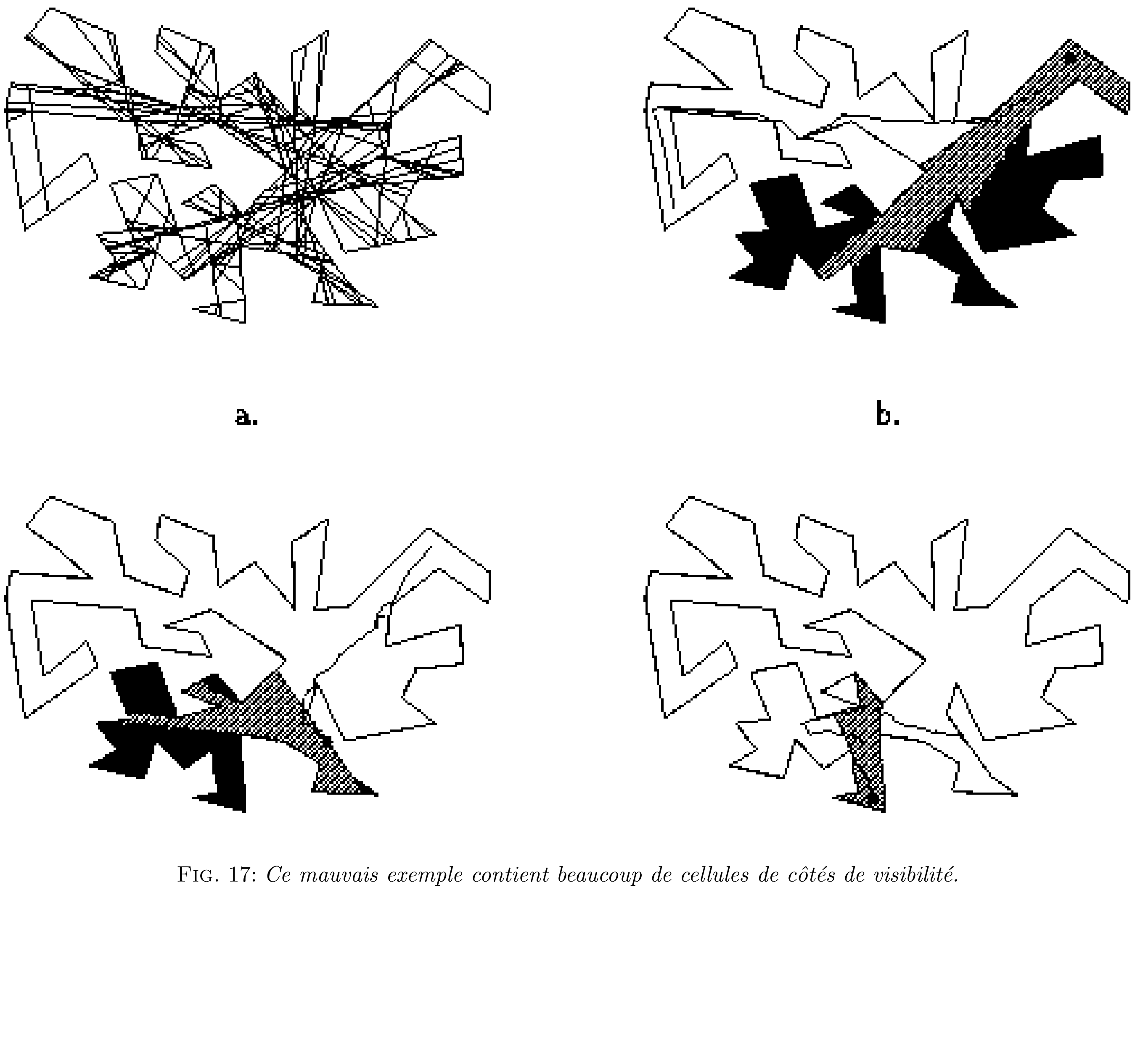

Fig. 17: *Ce mauvais exemple contient beaucoup de cellules de côtés de visibilité.*