

Connaissance a priori et oracles/conseils

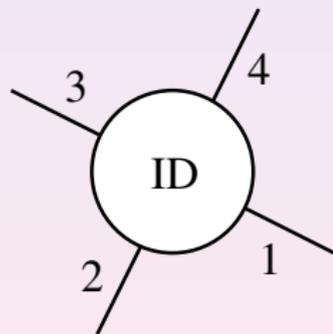
David ILCINKAS

CNRS & Université de Bordeaux (LaBRI)

Réunion Displexity
14 juin 2012

Types de connaissance

Connaissance **locale**



Connaissance **globale**

- nombre de sommets
- diamètre
- degré maximum
- genre, maille, treewidth, etc.

Observation

La **qualité/existence** des solutions algorithmiques **dépendent** souvent de la disponibilité de **connaissance globale** sur le réseau.

- [Lynch, PODC 1989] : A hundred impossibility proofs for distributed computing.
- [Fich, Ruppert, Distributed Computing, 2003] : Hundreds of impossibility results for distributed computing.

Quelques exemples

Exploration des graphes orientés en temps polynomial [BF+02]

- Impossible si aucune information
- Possible si connaissance d'une borne sup. sur n

Diffusion déterministe synchrone dans les réseaux radio

- Temps $\Omega(n \log D)$ si aucune information [CMS01]
- Temps $O(D + \log^2 n)$ si connaissance complète [KP06]

Réveil (wakeup) dans les réseaux filaires [AGPV90]

Connaissance du voisinage à distance ρ

- $O(n^{1+\Theta(1)/\rho})$ messages de taille bornée

Quelques exemples

Exploration des graphes orientés en temps polynomial [BF+02]

- Impossible si aucune information
- Possible si connaissance d'une borne sup. sur n

Diffusion déterministe synchrone dans les réseaux radio

- Temps $\Omega(n \log D)$ si aucune information [CMS01]
- Temps $O(D + \log^2 n)$ si connaissance complète [KP06]

Réveil (wakeup) dans les réseaux filaires [AGPV90]

Connaissance du voisinage à distance ρ

- $O(n^{1+\Theta(1)/\rho})$ messages de taille bornée

Quelques exemples

Exploration des graphes orientés en temps polynomial [BF+02]

- Impossible si aucune information
- Possible si connaissance d'une borne sup. sur n

Diffusion déterministe synchrone dans les réseaux radio

- Temps $\Omega(n \log D)$ si aucune information [CMS01]
- Temps $O(D + \log^2 n)$ si connaissance complète [KP06]

Réveil (wakeup) dans les réseaux filaires [AGPV90]

Connaissance du voisinage à distance ρ

- $O(n^{1+\Theta(1)/\rho})$ messages de taille bornée

Sur la nécessité d'une connaissance a priori

[A. Korman, J.-S. Sereni and L.Viennot. Toward more Localized Local Algorithms : Removing Assumptions concerning Global Knowledge. PODC'2011]

Résumé

Le papier présente une méthode plutôt générale pour transformer un algorithme de telle manière qu'il ne nécessite plus de connaissance globale connue a priori.

- l'algorithme transformé conserve la même complexité en temps asymptotique
- la méthode fonctionne pour de nombreux algorithmes déterministes et probabilistes ; en particulier MIS, Maximal Matching, coloring...

Sur la nécessité d'une connaissance a priori

[A. Korman, J.-S. Sereni and L.Viennot. Toward more Localized Local Algorithms : Removing Assumptions concerning Global Knowledge. PODC'2011]

Résultat

Le papier présente une **méthode plutôt générale** pour transformer un algorithme de telle manière qu'il **ne nécessite plus de connaissance globale** connue a priori.

- l'algorithme transformé conserve la **même complexité en temps** asymptotique
- la méthode fonctionne pour de nombreux algorithmes déterministes et probabilistes ; en particulier **MIS, Maximal Matching, coloring...**

Cas d'étude : exploration de graphes avec terminaison

Exploration de graphes...

Une entité mobile (robot/agent) doit **visiter** au moins une fois chaque sommet d'un graphe **anonyme**.

avec terminaison

L'entité mobile doit s'arrêter au bout d'un temps fini, i.e. détecter la terminaison.

Anonyme :

- sommets sans identifiant
- étiquetage local des arêtes (numéros de port)

Cas d'étude : exploration de graphes avec terminaison

Exploration de graphes...

Une entité mobile (robot/agent) doit **visiter** au moins une fois chaque sommet d'un graphe **anonyme**.

...avec terminaison

L'entité mobile doit **s'arrêter** au bout d'un temps fini, i.e. **détecter la terminaison**.

Anonyme :

- sommets sans identifiant
- étiquetage local des arêtes (numéros de port)

Cas d'étude : exploration de graphes avec terminaison

Exploration de graphes...

Une entité mobile (robot/agent) doit **visiter** au moins une fois chaque sommet d'un graphe **anonyme**.

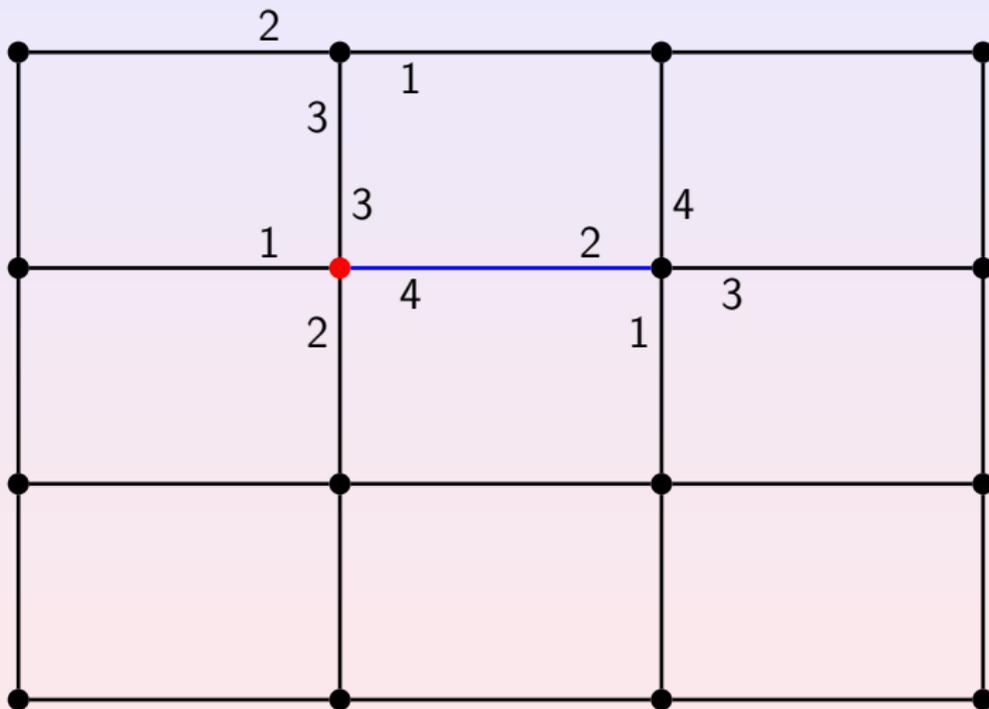
...avec terminaison

L'entité mobile doit **s'arrêter** au bout d'un temps fini, i.e. **détecter la terminaison**.

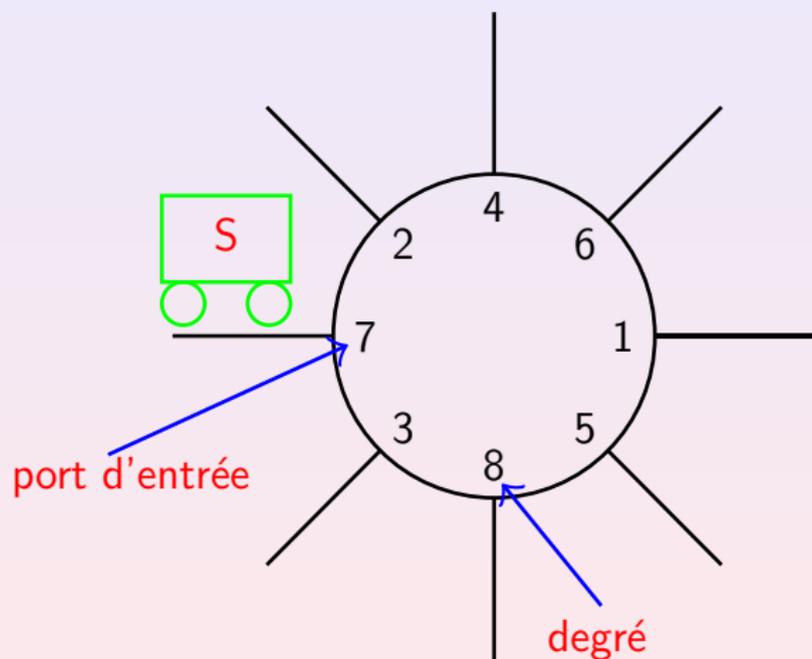
Anonyme :

- **sommets sans identifiant**
- étiquetage local des arêtes (numéros de port)

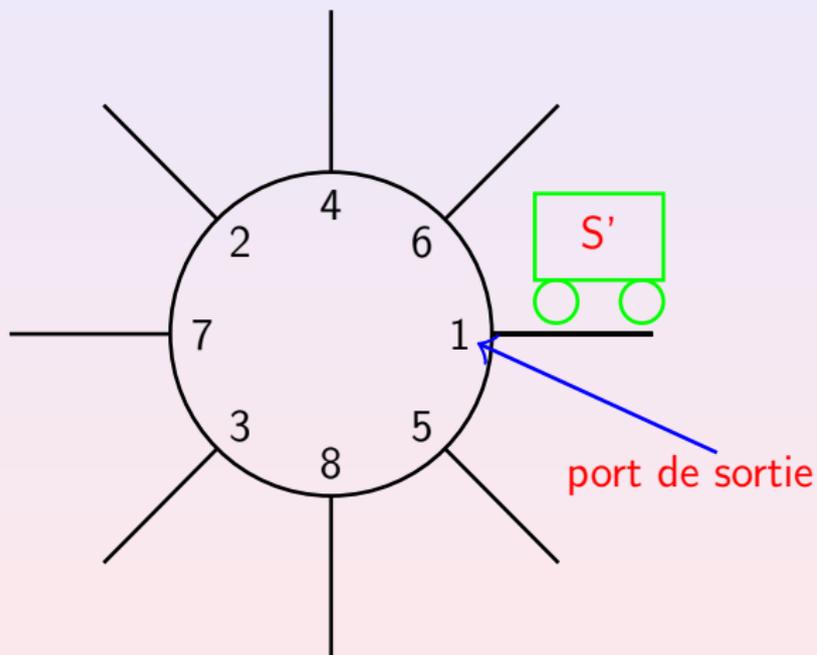
Exemple d'un graphe anonyme



Modèle de l'agent (1)



Modèle de l'agent (1)



Modèle de l'agent (2)

Entrées

- S : état courant
- i : numéro de port d'entrée
- d : degré du sommet

Fonction de transition

$$(S', j) = \mu (S, i, d)$$

Sorties

- S' : nouvel état
- j : numéro de port de sortie

Résultat positif

Théorème

Il existe un agent **capable de résoudre le problème d'exploration avec terminaison** s'il connaît une borne supérieure \hat{n} sur la taille n du graphe à explorer.

Preuve

Explorer tous les chemins de longueur \hat{n} partant du sommet de départ puis s'arrêter.

Corollaire

La connaissance d'une borne supérieure sur n est suffisante pour résoudre l'exploration avec terminaison.

Cette connaissance est-elle nécessaire ?

Résultat positif

Théorème

Il existe un agent **capable de résoudre le problème d'exploration avec terminaison** s'il connaît une borne supérieure \hat{n} sur la taille n du graphe à explorer.

Preuve

Explorer tous les **chemins de longueur \hat{n}** partant du sommet de départ puis s'arrêter.

Corollaire

La connaissance d'une borne supérieure sur n est suffisante pour résoudre l'exploration avec terminaison.

Cette connaissance est-elle nécessaire ?

Résultat positif

Théorème

Il existe un agent **capable de résoudre le problème d'exploration avec terminaison** s'il connaît une borne supérieure \hat{n} sur la taille n du graphe à explorer.

Preuve

Explorer tous les **chemins de longueur \hat{n}** partant du sommet de départ puis s'arrêter.

Corollaire

La **connaissance d'une borne supérieure** sur n est **suffisante** pour résoudre l'exploration avec terminaison.

Cette connaissance est-elle nécessaire ?

Résultat positif

Théorème

Il existe un agent **capable de résoudre le problème d'exploration avec terminaison** s'il connaît une borne supérieure \hat{n} sur la taille n du graphe à explorer.

Preuve

Explorer tous les **chemins de longueur \hat{n}** partant du sommet de départ puis s'arrêter.

Corollaire

La **connaissance d'une borne supérieure** sur n est **suffisante** pour résoudre l'exploration avec terminaison.

Cette connaissance est-elle **nécessaire** ?

Résultat négatif

Théorème

La **connaissance d'une borne supérieure** sur n est **nécessaire** pour résoudre l'exploration avec terminaison.

Preuve

- Supposons qu'il existe un agent explorant sans connaissance d'une borne supérieure sur n .
- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Êtes-vous convaincu ?

Est-ce une preuve satisfaisante ? Non : parité de n .

Résultat négatif

Théorème

La **connaissance d'une borne supérieure** sur n est **nécessaire** pour résoudre l'exploration avec terminaison.

Preuve

- Supposons qu'il existe un agent explorant **sans connaissance d'une borne supérieure** sur n .
- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Êtes-vous convaincu ?

Est-ce une preuve satisfaisante ? Non : parité de n .

Résultat négatif

Théorème

La **connaissance d'une borne supérieure** sur n est **nécessaire** pour résoudre l'exploration avec terminaison.

Preuve

- Supposons qu'il existe un agent explorant **sans connaissance d'une borne supérieure** sur n .
- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Êtes-vous **convaincu** ?

Est-ce une preuve satisfaisante? Non : parité de n .

Résultat négatif

Théorème

La **connaissance d'une borne supérieure** sur n est **nécessaire** pour résoudre l'exploration avec terminaison.

Preuve

- Supposons qu'il existe un agent explorant **sans connaissance d'une borne supérieure** sur n .
- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Êtes-vous **convaincu** ?

Est-ce une **preuve satisfaisante** ? Non : parité de n .

Résultat négatif

Théorème

La **connaissance d'une borne supérieure** sur n est **nécessaire** pour résoudre l'exploration avec terminaison.

Preuve

- Supposons qu'il existe un agent explorant **sans connaissance d'une borne supérieure** sur n .
- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Êtes-vous **convaincu** ?

Est-ce une **preuve satisfaisante** ? Non : parité de n .

Tentative de formalisation

Information a priori / conseil

- L'agent doit explorer le graphe G .
- **Au début** de l'exécution de l'algorithme, l'agent reçoit un **conseil** (connaissance a priori) d'un oracle \mathcal{O} sous la forme d'une **chaîne binaire** $\mathcal{O}(G)$.

\mathcal{O} induit une partition (F_0, F_1, \dots) de la famille de graphes \mathcal{G} :
 $F_i = \{G \in \mathcal{G} \mid \mathcal{O}(G) = i\}$

Connaitre une borne supérieure sur n

$$\forall i \exists \hat{n}_i, \forall G \in F_i \quad n(G) \leq \hat{n}_i$$

Tentative de formalisation

Information a priori / conseil

- L'agent doit explorer le graphe G .
- **Au début** de l'exécution de l'algorithme, l'agent reçoit un **conseil** (connaissance a priori) d'un oracle \mathcal{O} sous la forme d'une **chaîne binaire** $\mathcal{O}(G)$.

\mathcal{O} induit une **partition** (F_0, F_1, \dots) de la famille de graphes \mathcal{G} :
 $F_i = \{G \in \mathcal{G} \mid \mathcal{O}(G) = i\}$

Connaître une borne supérieure sur n

$$\forall i \exists \hat{n}_i, \forall G \in F_i \quad n(G) \leq \hat{n}_i$$

Tentative de formalisation

Information a priori / conseil

- L'agent doit explorer le graphe G .
- **Au début** de l'exécution de l'algorithme, l'agent reçoit un **conseil** (connaissance a priori) d'un oracle \mathcal{O} sous la forme d'une **chaîne binaire** $\mathcal{O}(G)$.

\mathcal{O} induit une **partition** (F_0, F_1, \dots) de la famille de graphes \mathcal{G} :
 $F_i = \{G \in \mathcal{G} \mid \mathcal{O}(G) = i\}$

Connaître une borne supérieure sur n

$$\forall i \exists \hat{n}_i \forall G \in F_i \quad n(G) \leq \hat{n}_i$$

Résultat négatif (bis)

Théorème

Un agent **sans connaissance d'une borne supérieure** sur la taille ne peut pas explorer **tous les cycles**.

(Rappel : conn. borne supérieure $\forall i \exists \hat{n}_i \forall G \in F_i \quad n(G) \leq \hat{n}_i$)

Preuve

- Supposons qu'il existe un agent explorant sans connaissance d'une borne supérieure sur n .
- $\exists i \forall N \exists G_N^{(i)} \in F_i \quad n(G_N^{(i)}) > N$
- Agent sur $G_1^{(i)}$: arrêt au temps k
- Agent sur $G_k^{(i)}$: arrêt avant l'exploration

Résultat négatif (bis)

Théorème

Un agent **sans connaissance d'une borne supérieure** sur la taille ne peut pas explorer **tous les cycles**.

(Rappel : conn. borne supérieure $\forall i \exists \hat{n}_i \forall G \in F_i \quad n(G) \leq \hat{n}_i$)

Preuve

- Supposons qu'il existe un agent explorant **sans connaissance d'une borne supérieure** sur n .
- $\exists i \forall N \exists G_N^{(i)} \in F_i \quad n(G_N^{(i)}) > N$
- Agent sur $G_1^{(i)}$: arrêt au temps k
- Agent sur $G_k^{(i)}$: arrêt avant l'exploration

Fin de l'histoire ?

Ensemble des cycles \subset ensemble des graphes

Corollaire

La connaissance d'une borne supérieure sur la taille est également nécessaire pour explorer tous les graphes.

FAUX!

Pas de monotonie...

$\mathcal{O}(G) = 0$ si G est un chemin et $n(G)$ sinon

Fin de l'histoire ?

Ensemble des cycles \subset ensemble des graphes

Corollaire

La connaissance d'une borne supérieure sur la taille est également nécessaire pour explorer tous les graphes.

FAUX!

Pas de monotonie...

$\mathcal{O}(G) = 0$ si G est un chemin et $n(G)$ sinon

Comparaison entre oracles/conseils

Idée : s'abstraire des infos que l'agent peut **calculer seul**

Définition : $\mathcal{O} \succeq \mathcal{O}'$

Il existe un algorithme d'agent mobile \mathcal{A} tel que $\mathcal{O} + \mathcal{A}$ peut **simuler** la sortie de \mathcal{O}' .

Théorème (nouvel essai)

Pour tout \mathcal{O} permettant de résoudre l'exploration dans tous les graphes, il existe \mathcal{O}' donnant une borne sup, tel que $\mathcal{O} \succeq \mathcal{O}'$.

Preuve

\mathcal{O} et l'algo d'exploration correspondant $\mathcal{A}_{\mathcal{O}}$ donnent une borne supérieure sur la taille de G : le temps de terminaison de $\mathcal{A}_{\mathcal{O}}$.

Comparaison entre oracles/conseils

Idée : s'abstraire des infos que l'agent peut **calculer seul**

Définition : $\mathcal{O} \succeq \mathcal{O}'$

Il existe un algorithme d'agent mobile \mathcal{A} tel que $\mathcal{O} + \mathcal{A}$ peut **simuler** la sortie de \mathcal{O}' .

Théorème (nouvel essai)

Pour tout \mathcal{O} permettant de résoudre l'exploration dans tous les graphes, il existe \mathcal{O}' donnant une borne sup, tel que $\mathcal{O} \succeq \mathcal{O}'$.

Preuve

\mathcal{O} et l'algo d'exploration correspondant $\mathcal{A}_{\mathcal{O}}$ donnent une borne supérieure sur la taille de G : le temps de terminaison de $\mathcal{A}_{\mathcal{O}}$.

Comparaison entre oracles/conseils

Idée : s'abstraire des infos que l'agent peut **calculer seul**

Définition : $\mathcal{O} \succeq \mathcal{O}'$

Il existe un algorithme d'agent mobile \mathcal{A} tel que $\mathcal{O} + \mathcal{A}$ peut **simuler** la sortie de \mathcal{O}' .

Théorème (nouvel essai)

Pour tout \mathcal{O} permettant de résoudre l'exploration dans tous les graphes, il existe \mathcal{O}' donnant une borne sup, tel que $\mathcal{O} \succeq \mathcal{O}'$.

Preuve

\mathcal{O} et l'algo d'exploration correspondant $\mathcal{A}_{\mathcal{O}}$ donnent une borne supérieure sur la taille de G : le **temps de terminaison** de $\mathcal{A}_{\mathcal{O}}$.

Fin de l'histoire ? (bis)

Corollaire

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les graphes**.

"Problème" 1 : Peut-il y avoir plusieurs classes minimales ?

"Problème" 2 :

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les chemins**.

Fin de l'histoire ? (bis)

Corollaire

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les graphes**.

"Problème" 1 : Peut-il y avoir **plusieurs** classes minimales ?

"Problème" 2

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les chemins**.

Fin de l'histoire ? (bis)

Corollaire

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les graphes**.

"Problème" 1 : Peut-il y avoir **plusieurs** classes minimales ?

"Problème" 2

La **classe d'équivalence** de tout oracle donnant la connaissance d'une **borne supérieure** sur n est **minimale** pour le problème de l'exploration de **tous les chemins**.

Retour au point de départ

Théorème

L'**oracle vide/neutre** est strictement **plus faible** que tout **oracle d'exploration** pour la famille de tous les graphes.

Preuve

- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Ne peut-on rien dire de plus ?

Théorème

Tout oracle induisant un nombre fini de parts F_i est strictement plus faible que tout oracle d'exploration pour la famille de tous les graphes.

Retour au point de départ

Théorème

L'**oracle vide/neutre** est strictement **plus faible** que tout **oracle d'exploration** pour la famille de tous les graphes.

Preuve

- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Ne peut-on rien dire de plus ?

Théorème

Tout oracle induisant un nombre fini de parts F_i est strictement plus faible que tout oracle d'exploration pour la famille de tous les graphes.

Retour au point de départ

Théorème

L'**oracle vide/neutre** est strictement **plus faible** que tout **oracle d'exploration** pour la famille de tous les graphes.

Preuve

- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Ne peut-on rien dire de plus ?

Théorème

Tout oracle induisant un nombre fini de parts F_i est strictement plus faible que tout oracle d'exploration pour la famille de tous les graphes.

Retour au point de départ

Théorème

L'**oracle vide/neutre** est strictement **plus faible** que tout **oracle d'exploration** pour la famille de tous les graphes.

Preuve

- Agent sur un triangle : arrêt au temps k
- Agent sur cycle C_{2k} : arrêt avant l'exploration

Ne peut-on rien dire de plus ?

Théorème

Tout **oracle induisant un nombre fini de parts F_i** est strictement **plus faible** que tout **oracle d'exploration** pour la famille de tous les graphes.

Définition

Le graphe H est un **quasi-revêtement de rayon r** du graphe G s'il existe un sommet u de H telle que la boule dans H de centre u et de rayon r est **indistinguable** de G .

Caractérisation

Un oracle permet de résoudre l'exploration dans une famille \mathcal{F} si et seulement si il n'existe aucune part F_i de \mathcal{F} comprenant des quasi-revêtements de rayon non-borné d'un même graphe $G \in F_i$.

Voir [J. Chalopin, E. Godard and Y. Métivier. Local Terminations and Distributed Computability in Anonymous Networks. Distributed Computing 2008]

Caractérisation fine

Définition

Le graphe H est un **quasi-revêtement de rayon r** du graphe G s'il existe un sommet u de H telle que la boule dans H de centre u et de rayon r est **indistinguable** de G .

Caractérisation

Un oracle permet de résoudre l'exploration dans une famille \mathcal{F} si et seulement si il n'existe aucune part F_i de \mathcal{F} comprenant des quasi-revêtements de rayon non-borné d'un même graphe $G \in F_i$.

Voir [J. Chalopin, E. Godard and Y. Métivier, Local Terminations and Distributed Computability in Anonymous Networks. Distributed Computing 2008]

Caractérisation fine

Définition

Le graphe H est un **quasi-revêtement de rayon r** du graphe G s'il existe un sommet u de H telle que la boule dans H de centre u et de rayon r est **indistinguable** de G .

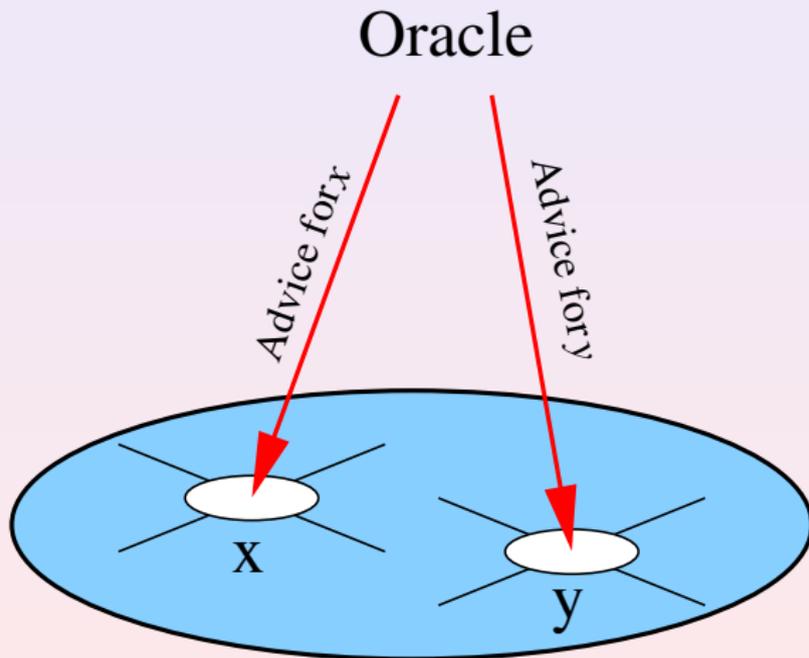
Caractérisation

Un oracle permet de résoudre l'exploration dans une famille \mathcal{F} si et seulement si il n'existe aucune part F_i de \mathcal{F} comprenant des quasi-revêtements de rayon non-borné d'un même graphe $G \in F_i$.

Voir [J. Chalopin, E. Godard and Y. Métivier. Local Terminations and Distributed Computability in Anonymous Networks. Distributed Computing 2008]

Connaissance globale donnée par un oracle

Cadre général en algorithmique distribuée :



Algorithmique distribuée avec conseil

Conception de deux éléments

l'oracle \mathcal{O}

- prend la totalité du graphe G en entrée
- fournit un conseil $\mathcal{O}(G, x)$ à chaque sommet (ou agent) x

l'algorithme \mathcal{A}

- **exécuté localement** par chaque sommet (ou agent)
- **utilise** les **conseils** donnés par l'oracle

Approches qualitative et quantitative

Inconvénient de l'approche qualitative

Difficile de comparer

- Algorithme connaissant n
- Algorithme connaissant D
- Algorithme connaissant le voisinage

Approche quantitative

Quelle est la taille minimum d'un oracle permettant de résoudre un problème \mathcal{P} ?

Questions quantitatives sur la connaissance nécessaire, indépendamment du type de connaissance fourni.

Approches qualitative et quantitative

Inconvénient de l'approche qualitative

Difficile de comparer

- Algorithme connaissant n
- Algorithme connaissant D
- Algorithme connaissant le voisinage

Approche quantitative

Quelle est la **taille minimum d'un oracle** permettant de résoudre un problème \mathcal{P} ?

Questions **quantitatives** sur la connaissance nécessaire, indépendamment du **type** de connaissance fourni.

Application au calcul distribué

Deux tâches fondamentales

Dissémination d'un message M d'une source à tous les nœuds d'un réseau.

- **Réveil (Wakeup)** : un nœud ne peut pas communiquer avant d'avoir reçu le message M
- **Diffusion (Broadcast)** : un nœud peut communiquer à tout instant

Problème

Résoudre ces tâches en utilisant un nombre de messages linéaire en n

Application au calcul distribué

Deux tâches fondamentales

Dissémination d'un message M d'une source à tous les nœuds d'un réseau.

- Réveil (Wakeup) : un nœud ne peut pas communiquer avant d'avoir reçu le message M
- Diffusion (Broadcast) : un nœud peut communiquer à tout instant

Problème

Résoudre ces tâches en utilisant un nombre de messages linéaire en n

[P. Fraigniaud, D. Ilcinkas, and A. Pelc. *Communication algorithms with advice*. JCSS 2010]

Réveil (Wakeup)

La taille minimum d'oracle est $\Theta(n \log n)$ bits.

Diffusion (Broadcast)

La taille minimum d'oracle est $\Theta(n)$ bits.

Permet de distinguer clairement les deux problèmes.

[P. Fraigniaud, D. Ilcinkas, and A. Pelc. *Communication algorithms with advice*. JCSS 2010]

Réveil (Wakeup)

La taille minimum d'oracle est $\Theta(n \log n)$ bits.

Diffusion (Broadcast)

La taille minimum d'oracle est $\Theta(n)$ bits.

Permet de distinguer clairement les deux problèmes.

[P. Fraigniaud, D. Ilcinkas, and A. Pelc. *Communication algorithms with advice*. JCSS 2010]

Réveil (Wakeup)

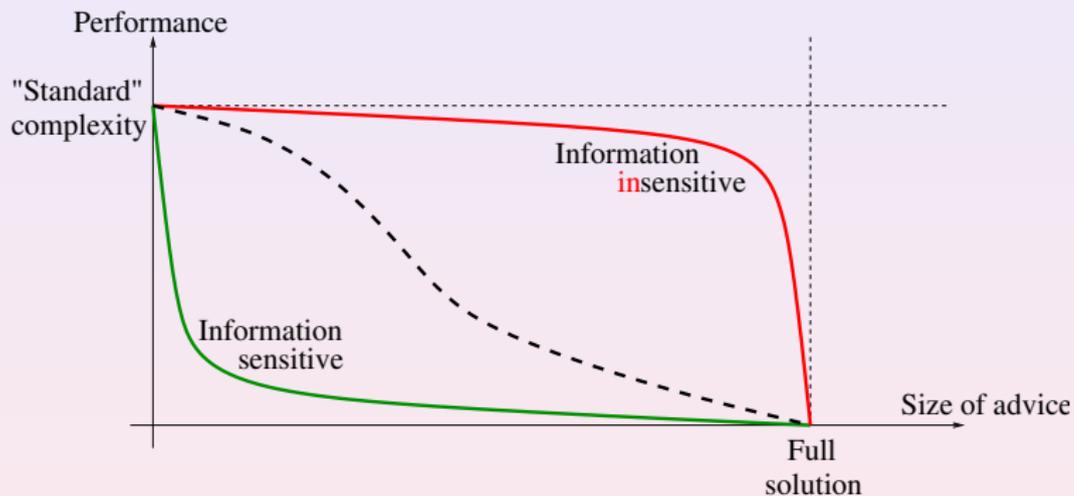
La taille minimum d'oracle est $\Theta(n \log n)$ bits.

Diffusion (Broadcast)

La taille minimum d'oracle est $\Theta(n)$ bits.

Permet de **distinguer** clairement les deux problèmes.

Autre point de vue possible



Merci
de votre attention