

Comment trouver des problèmes difficiles faciles ?

LAMSADE

Florian Sikora

Plan

Contourner la difficulté

Étude de cas : Graph Motif

Difficulté

Théorème du doigt mouillé

La plupart des problèmes combinatoires intéressants sont NP-difficiles

- Approuvé par F. Foch : *“Ne me dites pas que ce problème est difficile. S’il n’était pas difficile, ce ne serait pas un problème.”*



Difficulté

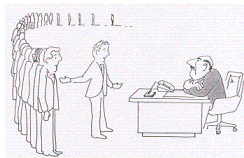
Théorème du doigt mouillé

La plupart des problèmes combinatoires intéressants sont NP-difficiles

- ▶ Approuvé par F. Foch : *“Ne me dites pas que ce problème est difficile. S’il n’était pas difficile, ce ne serait pas un problème.”*
- ▶ Problèmes **NP-complets** :
 - ▶ N'existe probablement **pas d'algorithme exact** de complexité pire cas **polynomiale** (par ex. $\mathcal{O}(n^c)$).

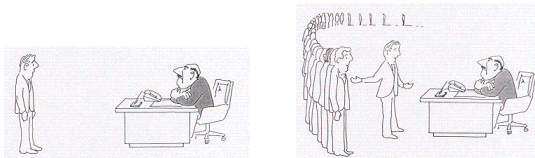
Difficulté

- Garey and Johnson 1979.
 - ▶ “Je suis trop bête pour trouver un algorithme efficace pour ce problème...”
 - ▶ “... comme toutes ces personnes reconnues !”



Difficulté

- Garey and Johnson 1979.
 - ▶ “Je suis trop bête pour trouver un algorithme efficace pour ce problème...”
 - ▶ “... comme toutes ces personnes reconnues !”



- Adaptation D. Hermelin 2009.
 - ▶ “Et alors ?! J’ai besoin d’une solution !”



Que faire ?

- ▶ Pour contourner la difficulté, boîte à outil (©F.H.) :
 - ▶ Trouver des instances plus simples.
 - ▶ Complexité paramétrée.
 - ▶ Algorithmes modérément exponentiels.
 - ▶ Approximation (polynomiale).
 - ▶ Approximation non polynomiale.
 - ▶ ILP.
 - ▶ ...

Complexité paramétrée

“Mesurer la complexité seulement en fonction de la taille de la donnée signifie ignorer toute information structurelle sur l'instance donnée...”

J. Flum et M. Grohe

“Question : Quand un problème venant de la “vie réelle” n'a pas d'autre structure que sa taille ?

Réponse : Jamais !”

R. Downey et M. Fellows

Complexité paramétrée

“Mesurer la complexité seulement en fonction de la taille de la donnée signifie ignorer toute information structurelle sur l'instance donnée...”

J. Flum et M. Grohe

“Question : Quand un problème venant de la “vie réelle” n'a pas d'autre structure que sa taille ?

Réponse : Jamais !”

R. Downey et M. Fellows

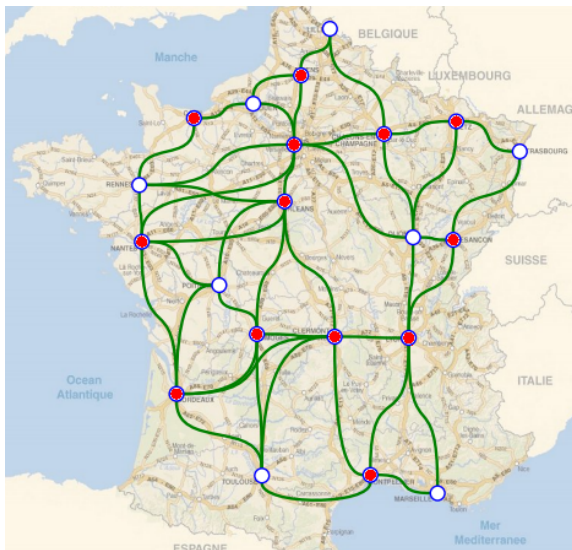
“L'idée fondamentale est de restreindre l'explosion combinatoire, semble-t-il inévitable, qui est responsable de la croissance exponentielle du temps de calcul, à un paramètre spécifique au problème...”

R. Niedermeier

Complexité paramétrée : Exemple du Vertex Cover



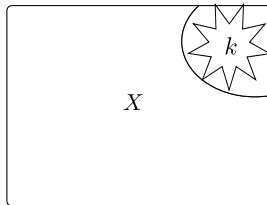
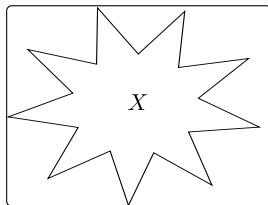
Complexité paramétrée : Exemple du Vertex Cover



Complexité paramétrée

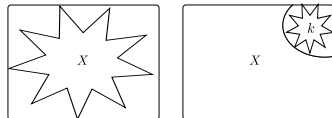


- Problème dans la classe FPT si algorithme **exact** avec complexité pire cas de la forme $f(k) \cdot |X|^c$.
 - Par ex. $O(2^k \cdot n^2)$ ou $O(2^{2^{2^{2^{2^k}}}} \cdot n)$.



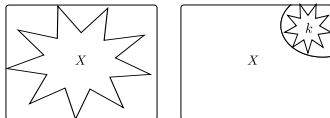
Complexité paramétrée - En pratique ?

- ▶ Souvent des paramètres naturels (petits) dans des applications.
 - ▶ Taille k de la requête dans une **base de données** de taille n .
 - ▶ k étapes dans un **jeu** à n mouvements possibles.
 - ▶ Aligner k **séquences d'ADN** de taille n .
 - ▶ Nombre k de sommets intéressants dans un **réseau social** de taille n .
 - ▶ Nombre n de **votants** parmi k candidats.
 - ▶ IA. **Strip Planning** : taille k du plan.
 - ▶ ...



Complexité paramétrée - En pratique ?

- ▶ Souvent des paramètres naturels (petits) dans des applications.
 - ▶ Taille k de la requête dans une **base de données** de taille n .
 - ▶ k étapes dans un **jeu** à n mouvements possibles.
 - ▶ Aligner k **séquences d'ADN** de taille n .
 - ▶ Nombre k de sommets intéressants dans un **réseau social** de taille n .
 - ▶ Nombre n de **votants** parmi k candidats.
 - ▶ IA. **Strip Planning** : taille k du plan.
 - ▶ ...



- ▶ $k \simeq 10000$ pour le problème CLUSTER EDITING en quelques minutes.

Complexité paramétrée - Dark side

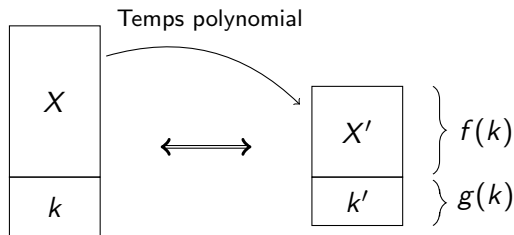
- ▶ Classes de complexités...
- ▶ **fpt-réductions** préservant le paramètre.
 - ▶ Transformer une instance (G, k) d'INDEPENDENT SET en une instance $(G, n - k)$ de VERTEX COVER **n'est pas** une fpt-réduction.
 - ▶ Transformer une instance (G, k) d'INDEPENDENT SET en une instance $(\overline{G}, k)$ de CLIQUE **est** une fpt-réduction.

Complexité paramétrée - Dark side

- ▶ Classes de complexités...
- ▶ **fpt-réductions** préservant le paramètre.
 - ▶ Transformer une instance (G, k) d'INDEPENDENT SET en une instance $(G, n - k)$ de VERTEX COVER **n'est pas** une fpt-réduction.
 - ▶ Transformer une instance (G, k) d'INDEPENDENT SET en une instance $(\overline{G}, k)$ de CLIQUE **est** une fpt-réduction.
- ▶ Existe une analogie avec le théorème de Cook-Levin pour “démarrer”.
- ▶ Par ex., INDEPENDENT SET pas dans FPT pour le paramètre taille de la solution.

Noyaux

- “Pre-processing” de l’instance.
- Obtenir en temps polynomial une **instance équivalente** de **taille bornée** par une fonction du paramètre.
- But : noyau de la plus petite taille possible (polynomial vs non-polynomial)



Noyaux - Max 3-Sat

Max 3-Sat

- ▶ Entrée : n variables, m clauses.
- ▶ Paramètre : un entier k .
- ▶ Question : Existe-t-il une affectation des variables pour satisfaire au moins k clauses ?

Noyaux - Max 3-Sat

Max 3-Sat

- ▶ Entrée : n variables, m clauses.
 - ▶ Paramètre : un entier k .
 - ▶ Question : Existe-t-il une affectation des variables pour satisfaire au moins k clauses ?
-
- ▶ Il existe une affectation pour satisfaire la **moitié des clauses**.
 - ▶ Prendre n'importe quelle affectation :
 - ▶ Si elle satisfait la moitié des clauses, OK.
 - ▶ Sinon, son complémentaire satisfait la moitié des clauses, OK.
 - ▶ Si $k \leq m/2$: réduire à une instance OUI triviale (1 clause, $k = 1$).
 - ▶ Sinon, $m \leq 2k$, donc $n \leq 6k$: instance bornée par une fonction (linéaire) de k .

Noyaux - Max 3-Sat

Max 3-Sat

- ▶ Entrée : n variables, m clauses.
 - ▶ Paramètre : un entier k .
 - ▶ Question : Existe-t-il une affectation des variables pour satisfaire au moins k clauses ?
-
- ▶ Il existe une affectation pour satisfaire la **moitié des clauses**.
 - ▶ Prendre n'importe quelle affectation :
 - ▶ Si elle satisfait la moitié des clauses, OK.
 - ▶ Sinon, son complémentaire satisfait la moitié des clauses, OK.
 - ▶ Si $k \leq m/2$: réduire à une instance OUI triviale (1 clause, $k = 1$).
 - ▶ Sinon, $m \leq 2k$, donc $n \leq 6k$: instance bornée par une fonction (linéaire) de k .
 - ▶ Un noyau implique l'appartenance à la classe FPT pour ce paramètre (brute-force).

Algorithmes modérément exponentiels

- ▶ But : avoir un algorithme exact en $O(c^n)$, petit c !
 - ▶ Par ex., $O(1.8^n)$ à $O(1.7^n)$: facteur **multiplicatif** sur la taille des instances possibles.

Algorithmes modérément exponentiels

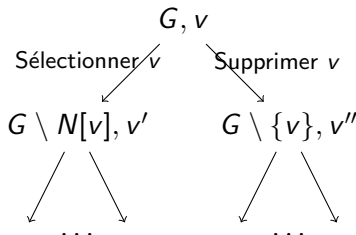
- ▶ MAXIMUM INDEPENDENT SET naïf :
 - ▶ Tester **tous les sous-ensembles** de sommets et renvoyer le plus grand.
 - ▶ $\Omega(2^n)$.

Branchements pour Maximum Independent Set

- ▶ Brancher sur deux sous-problèmes et résoudre récursivement.
- ▶ Pour chaque v , soit :
 - ▶ Sélectionner v (et supprimer son voisinage).
 - ▶ Supprimer v .

Branchements pour Maximum Independent Set

- ▶ Brancher sur deux sous-problèmes et résoudre récursivement.
- ▶ Pour chaque v , soit :
 - ▶ Sélectionner v (et supprimer son voisinage).
 - ▶ Supprimer v .
- ▶ Algo *MIS* récursif :
 - ▶ Si $\deg(G) \leq 2$: polynomial.
 - ▶ Sinon,
 - ▶ Choisir un v .
 - ▶ Retourner $\max(1 + \text{MIS}(G \setminus N[v]), \text{MIS}(G \setminus \{v\}))$.

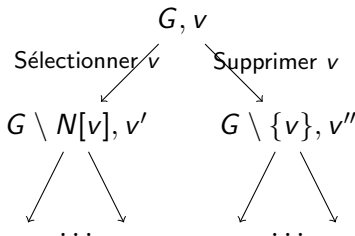


Branchements pour Maximum Independent Set

- ▶ Brancher sur deux sous-problèmes et résoudre récursivement.
- ▶ Pour chaque v , soit :
 - ▶ Sélectionner v (et supprimer son voisinage).
 - ▶ Supprimer v .

▶ Algo *MIS* récursif :

- ▶ Si $\deg(G) \leq 2$: polynomial.
- ▶ Sinon,
 - ▶ Choisir un v .
 - ▶ Retourner $\max(1 + \text{MIS}(G \setminus N[v]), \text{MIS}(G \setminus \{v\}))$.



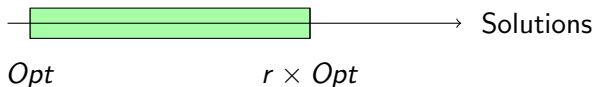
- ▶ $T(n) \leq T(n-1) + T(n-d(v)-1) \leq T(n-1) + T(n-4)$.
- ▶ Solution pour $x^n = x^{n-1} + x^{n-4}$.
- ▶ $O(1.3803^n \cdot \text{poly}(n)) = O^*(1.3803^n)$

Approximation

- ▶ Pour un problème d'optimisation NP-difficile, pas d'algorithme **polynomial** possible (sauf si $P = NP$)...
- ▶ ... si la solution doit être optimale !
- ▶ Sacrifier de l'exactitude pour obtenir un algorithme polynomial.

Approximation (problème de minimisation)

- ▶ On borne l'erreur.
- ▶ Un algorithme est de ***r*-approximation** s'il est polynomial et qu'il retourne des solutions dont le coût est au pire $r \times Opt$.



Approximation – Difficulté

- Possible de prouver qu'il n'existe pas d'algorithme d'approximation inférieur à un certain ratio.
- But : rapprocher les deux frontières.



Exemple pour VERTEX COVER

Approximation en temps exponentiel

- ▶ Problème MAX INDEPENDENT SET.
- ▶ On sait le résoudre en $O^*(c^n)$.
- ▶ Propriété héréditaire
 - ▶ Vérifie la propriété sur G ssi tous ses sous-graphes la vérifient.

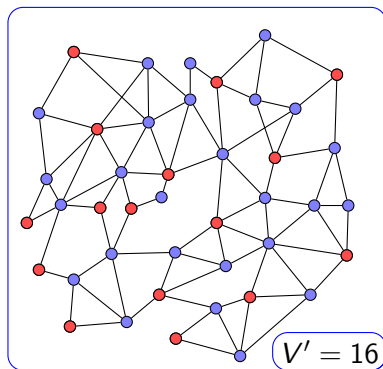


Figure E. Tournaire

Approximation en temps exponentiel

- ▶ Problème MAX INDEPENDENT SET.
- ▶ On sait le résoudre en $O^*(c^n)$.
- ▶ Propriété héréditaire
 - ▶ Vérifie la propriété sur G ssi tous ses sous-graphes la vérifient.
- ▶ Partager l'instance en 2 parts égales.
- ▶ Résoudre sur chaque partie.
- ▶ Renvoyer le plus grand tel quel.
- ▶ Ratio 1/2 : $O^*(c^{n/2})$.

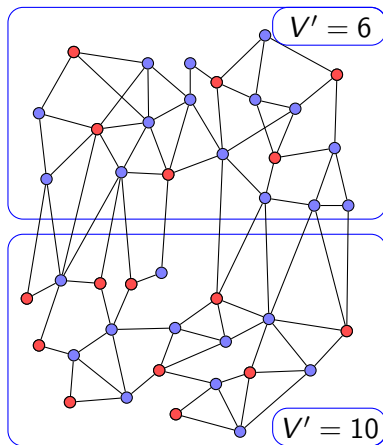


Figure E. Tournaire

Approximation en temps exponentiel

- ▶ Problème MAX INDEPENDENT SET.
- ▶ On sait le résoudre en $O^*(c^n)$.
- ▶ Propriété héréditaire
 - ▶ Vérifie la propriété sur G ssi tous ses sous-graphes la vérifient.
- ▶ Partager l'instance en 2 parts égales.
- ▶ Résoudre sur chaque partie.
- ▶ Renvoyer le plus grand tel quel.
- ▶ Ratio 1/2 : $O^*(c^{n/2})$.
- ▶ Pas de ratio meilleur que $n^{1-\epsilon}$ en temps polynomial.

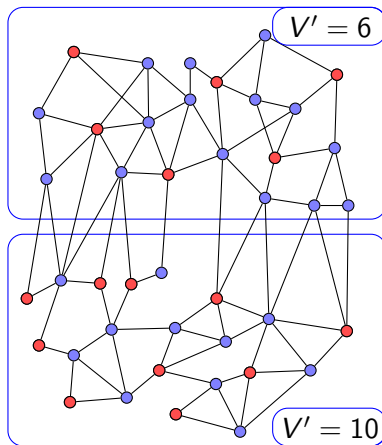


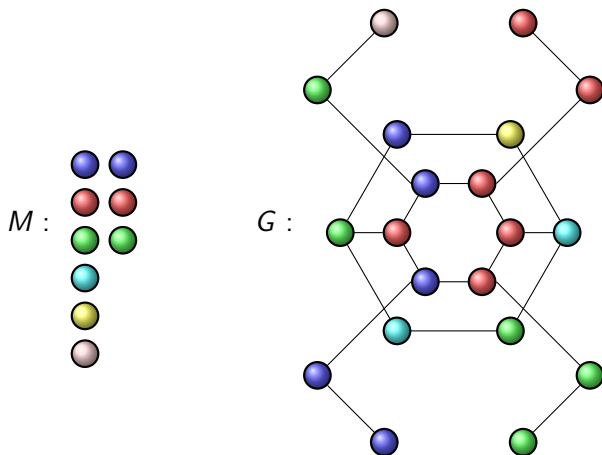
Figure E. Tournaire

Plan

Contourner la difficulté

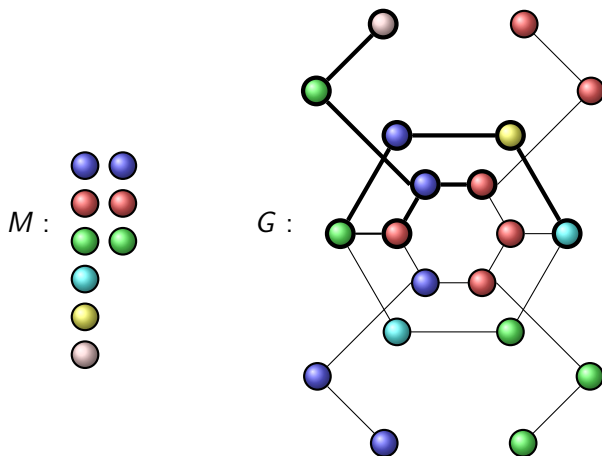
Étude de cas : Graph Motif

Graph Motif



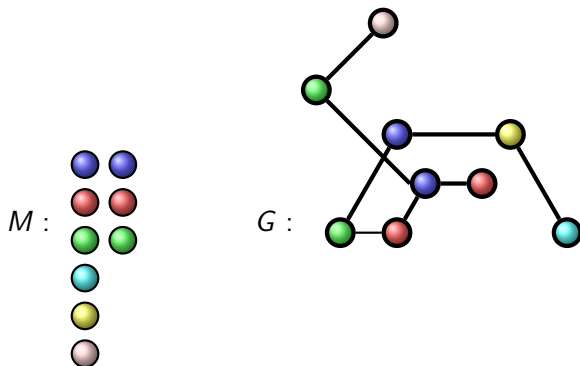
- Sortie : existe t-il un sous-graphe connexe contenant exactement toutes les couleurs de M ?

Graph Motif



- Sortie : existe t-il un sous-graphe connexe contenant exactement toutes les couleurs de M ?

Graph Motif



- Sortie : existe t-il un sous-graphe connexe contenant exactement toutes les couleurs de M ?

Graph Motif – Difficulté

- ▶ Le problème reste NP-complet, même si :
 - ▶ G est un arbre [LACROIX ET AL. 2006].
 - ▶ Cet **arbre est de degré maximum 3** et une occurrence par couleur dans M [FELLOWS ET AL. 2007].
 - ▶ L'**arbre est de profondeur 2** et une occurrence par couleur dans M [AMBALATH ET AL. 2010].
 - ▶ **2 couleurs** différentes dans M et G est un graphe biparti de degré maximum 4 [FELLOWS ET AL. 2007].

Graph Motif – Difficulté

- ▶ Le problème reste NP-complet, même si :
 - ▶ G est un arbre [LACROIX ET AL. 2006].
 - ▶ Cet **arbre est de degré maximum 3** et une occurrence par couleur dans M [FELLOWS ET AL. 2007].
 - ▶ L'**arbre est de profondeur 2** et une occurrence par couleur dans M [AMBALATH ET AL. 2010].
 - ▶ **2 couleurs** différentes dans M et G est un graphe biparti de degré maximum 4 [FELLOWS ET AL. 2007].
- ▶ Polynomial si :
 - ▶ G est un **caterpillar** (suppression des feuilles de l'arbre donne un chemin) [AMBALATH ET AL. 2010].
 - ▶ Moins de n^2 sous-chemins possibles.
 - ▶ G est un arbre où chaque couleur **apparaît au plus deux fois**, une occurrence par couleur dans M [S. 2011].
 - ▶ 2 SAT

Graph Motif – Complexité paramétré

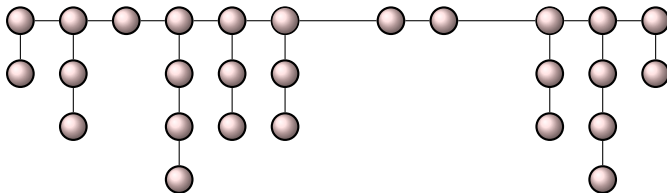
- ▶ Le problème est dans FPT avec le paramètre $k = |M|$.
“Course” :
 - ▶ $O^*(87^k)$ [FELLOWS ET AL. 2007].
 - ▶ $O^*(4.3^k)$ (programmation dynamique et technique du color-coding) [BETZLER ET AL. 2008].
 - ▶ $O^*(4^k)$ et espace polynomial (Recherche de monômes multilinéaires) [GUILLEMOT ET S. 2012].
 - ▶ $O^*(2.54^k)$ [KOUTIS 2012].
 - ▶ $O^*(2^k)$ et espace polynomial (Narrow sieves) [BJORKLUND ET AL. 2013] [PINTER ET AL. 2013].
 - ▶ $O((2 - \epsilon)^k)$ improbable [BJORKLUND ET AL. 2013].

Graph Motif – Complexité paramétré

- ▶ Le problème est dans FPT avec le paramètre $k = |M|$.
“Course” :
 - ▶ $O^*(87^k)$ [FELLOWS ET AL. 2007].
 - ▶ $O^*(4.3^k)$ (programmation dynamique et technique du color-coding) [BETZLER ET AL. 2008].
 - ▶ $O^*(4^k)$ et espace polynomial (Recherche de monômes multilinéaires) [GUILLEMOT ET S. 2012].
 - ▶ $O^*(2.54^k)$ [KOUTIS 2012].
 - ▶ $O^*(2^k)$ et espace polynomial (Narrow sieves) [BJORKLUND ET AL. 2013] [PINTER ET AL. 2013].
 - ▶ $O((2 - \epsilon)^k)$ improbable [BJORKLUND ET AL. 2013].
- ▶ Pas dans FPT pour le paramètre “nombre de couleurs différentes dans M ” [FELLOWS ET AL. 2007].

Graph Motif – Noyaux

- Pas de noyau polynomial même si G est un comb graph
[AMBALATH ET AL. 2010].

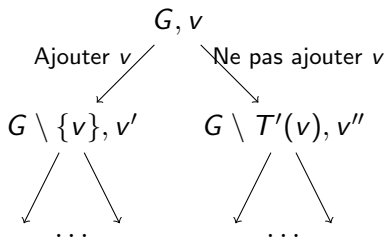


Graph Motif – Approximation

- ▶ Différentes versions d'optimisations associées.
- ▶ Par exemple, maximiser la taille de la solution.
 - ▶ Difficile à approcher.
 - ▶ APX-difficile, même pour G arbre de degré maximum 3 [DONDI ET AL. 2009].
 - ▶ Pas approximable à moins de $n^{1/3-\epsilon}$, même pour G arbre où chaque couleur apparaît au plus deux fois [RIZZI ET S. 2012].

Graph Motif – Algorithme modérément exponentiel

- ▶ Si G est un **arbre**, algorithme en $O^*(1.62^n)$ [DONDI ET AL. 2009].
- ▶ Suppose la racine dans la solution.
- ▶ Test de l'ajout de feuilles en temps polynomial.
- ▶ Branchement sur les sommets **internes**.
 1. L'ajouter dans la solution.
 2. Ne pas l'ajouter et supprimer son sous-arbre (donc au moins 2 sommets).



- ▶ $O^*(1.33^n)$ si chaque couleur n'apparaît qu'au plus une fois dans M .

