Université Assane SECK de Ziguinchor

Unité de Formation et de Recherche des Sciences et Technologies

Département d'Informatique

ROLE DES ALGORITHMES EN INFORMATIQUE

Licence 1 Math - Physique – Informatique

Janvier 2021

©Youssou DIENG

ydieng@univ-zig.sn

1 - LES ALGORITHMES

L'objectif de ce chapitre est de répondre aux interrogations suivantes: Qu'est-ce qu'un algorithme ? En quoi l'étude des algorithmes est-elle utile ? Quel est le rôle des algorithmes par rapport aux autres technologies informatiques ?

Définition : *Un algorithme est une procédure de calcul bien définie qui prend en entrée une valeur, ou un ensemble de valeurs, et qui donne en sortie une valeur, ou un ensemble de valeurs.*

Un algorithme est donc une séquence d'étapes de calcul qui transforment l'entrée en sortie. C'est un outil permettant de résoudre un problème de calcul bien spécifié. L'énoncé du problème spécifie, en termes généraux, la relation désirée entre l'entrée et la sortie. L'algorithme décrit une procédure de calcul spécifique permettant d'obtenir cette relation entrée/sortie.

1.1 - Exemple : Problème de tri

Input: A sequence of n numbers $\langle a_1, a_2, \ldots, a_n \rangle$.

Output: A permutation (reordering) $\langle a'_1, a'_2, \ldots, a'_n \rangle$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

En effet, si on considère la sequence A = "***31, 41, 59, 26, 41, 58****", alors un algotithme de trie produit en sortie la suite "****26, 31, 41, 41, 58, 59"****.* La suite donnée en entrée est appelée **instance** du problème de tri. Le tri est une opération majeure en informatique (phase intermédiaire de beaucoup de programme).

1.2 - Algorithme est dit correct :

Un algorithme est dit correct si, pour chaque instance en entrée, il se termine en produisant la bonne sortie. Un algorithme correct résout le problème donné. Un algorithme peut être spécifié en langage humain ou en langage informatique, mais peut aussi être basé sur un système matériel. L'unique obligation est que la spécification fournisse une description précise de la procédure de calcul à suivre.

1.3 - Quels sont les types de problème susceptibles d'être résolus par des algorithmes ?

1.1.3.1 - Le projet du génome humain

Dans le Projet di génome humain, on cherche à Identifier les 100 000 gènes de l'ADN humain, à déterminer les séquences des 3 milliards de paires de bases chimiques qui constituent l'ADN humain, à stocker ces informations dans des bases de données et à développer des outils d'analyse de données. Chacune de ces étapes exige des algorithmes très élaborés.

1.1.3.2 - INTERNET

Internet permet à des gens éparpillés un peu partout dans le monde d'accéder rapidement à toutes sortes de données. Tout cela repose sur des algorithmes intelligents qui permettent de gérer et manipuler de grosses masses de données. Comme exemples de problèmes à résoudre nous pouvons citer le problème de la recherche de routes optimales pour l'acheminement des données ou d'utiliser un moteur de recherche pour trouver rapidement les pages contenant tel ou tel type de données.

1.1.3.3 - LE COMMERCE ELECTRONIQUE

Le commerce électronique permet de négocier et échanger, de manière électronique, des biens et services. il exige que l'on préserve la confidentialité des données telles que numéros de carte de crédit, mots de passe, relevés bancaires.

La cryptographie à clé publique et les signatures numériques qui font partie des technologies fondamentales employées dans ce contexte, s'appuient sur des algorithmes numériques et sur la théorie des nombres.

1.1.3.4 - L'INDUSTRIE ET LE COMMERCE

Il est souvent question d'optimiser l'allocation de ressources limitées. Par exemples on peut prendre le cas d'une compagnie pétrolière qui veut savoir où placer ses puits de façon à maximiser les profits escomptés. Ou encore un candidat à la présidence veut savoir dans quels supports publicitaires il doit investir pour maximiser ses chances d'élection. Un autre exemple qu'on peut prendre est le cas d'une compagnie aérienne qui désire réaliser l'affectation des équipages aux vols de telle façon que les coûts soient minimisés, les vols assurés sans défaillance et la législation respectée. En fin le dernier cas est un fournisseur de services

Internet qui veut savoir où placer des ressources supplémentaires pour desservir ses clients de manière plus efficace.

1.4 - Structures de données

Une structures de données est un moyen de stocker et organiser des données pour faciliter l'accès à ces données et leur modification. Il n'y a aucune structure de données qui réponde à tous les besoins, de sorte qu'il importe de connaître les forces et limitations de plusieurs de ces structures.

1.5 - Problèmes difficiles

Notre mesure habituelle de l'efficacité est la vitesse, c'est-à-dire la durée que met un algorithme à produire ses résultats. Cependant, il existe des problèmes, pour lesquels on ne connaît aucune solution efficace. Dans le cours de Modèle de Calcul de Master on étudie un sous-ensemble intéressant de ces problèmes, connus sous l'appellation de problèmes **NP-complets**.

On n'a jamais trouvé d'algorithme efficace pour un problème NP-complet, mais personne n'a jamais prouvé qu'il ne peut pas exister d'algorithme efficace pour un problème. Autrement dit, on ne sait pas s'il existe ou non des algorithmes efficaces pour les problèmes NP-complets. L'ensemble des problèmes NP-complets offre la propriété remarquable suivante :

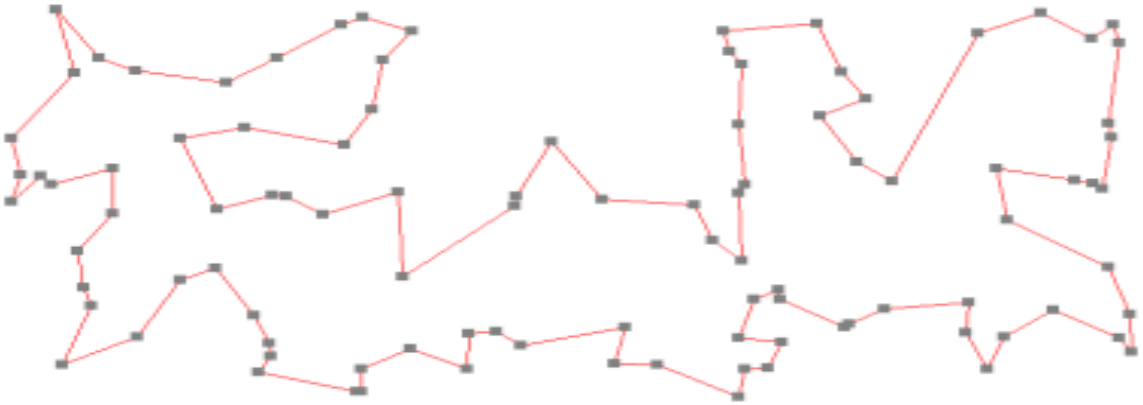
- s'il existe un algorithme efficace pour un quelconque de ces problèmes, alors il existe des algorithmes efficaces pour tous.
- Plusieurs problèmes NP-complets ressemblent, sans être identiques, à des problèmes pour lesquels nous connaissons des algorithmes efficaces.

1.6 - Exemple1 : Le Voyageur de commerce

Prenons le cas d'une entreprise de camionnage ayant un dépôt central. Chaque jour, elle charge le camion au dépôt puis l'envoie faire des livraisons à plusieurs endroits. À la fin de la journée, le camion doit revenir au dépôt de façon à pouvoir être rechargé le jour suivant.

Pour réduire les coûts, l'entreprise veut choisir un ordre de livraisons tel que la distance parcourue par le camion soit minimale. Ce problème, qui n'est autre que le fameux « problème

du voyageur de commerce », est un problème NP-complet. Il n'y a pas d'algorithme efficace connu.



Sous certaines hypothèses, cependant, il existe des algorithmes efficaces donnant une distance globale qui n'est pas trop éloignée de la distance minimale. Ce genre d'algorithmes sont dits « algorithmes d'approximation ».

*Exemple*2: ***Problème de la Somme d'entiers***

Soient donnés une suite finie d'entiers x1,...,xk et un entier s. Le problème qui est de savoir s'il est possible d'extraire une sous-suite de la suite donnée dont la somme est égal à s. En d'autres termes, il s'agit de trouver suite croissante d'indices $1 \leq i_1 < i_2 < \ldots < i_n \leq$ k telle que $x i_1 + x i_2 + \ldots + x i_n = s$.

2 - LES ALGORITHMES EN TANT QUE TECHNOLOGIE

Si on supposait que les ordinateurs soient infiniment rapides et que leurs mémoires soient gratuites. Alors, la question légitime à se poser est : faudrait-il étudier les algorithmes ?

La réponse est oui, ne serait-ce que pour montrer que la solution ne boucle pas indéfiniment et qu'elle se termine avec la bonne réponse. En effet, si les ordinateurs étaient infiniment rapides, alors toute méthode correcte de résolution d'un problème ferait l'affaire. Vous voudriez sans doute que votre solution entre dans le cadre d'une bonne méthodologie d'ingénierie (c'est-à-dire qu'elle soit bien conçue et bien documentée). Cependant, vous privilégierez le plus souvent la méthode qui est la plus simple à mettre en œuvre.

2.1 - Efficacité

Il arrive souvent que des algorithmes conçus pour résoudre le même problème diffèrent fortement entre eux en termes d'efficacité. Ces différences peuvent être bien plus importantes que celles dues au matériel et au logiciel.

En effet, l'algorithme du **Tri par insertion** prend un temps approximativement égal à $c_1 n^2$ pour trier n éléments, $c1$ étant une constante indépendante de n. Ce qui veut dire que la durée du tri est, grosso modo, proportionnelle à n^2.

Cependant l'algorithme du **Tri par fusion** prend un temps approximativement égal à $c2n \lg n$, où $\lg n$ désigne $\log_2 n$ et $c2$ est une autre constante indépendante, elle aussi, de n. Le tri par insertion a généralement un facteur constant inférieur à celui du tri par fusion, de sorte que $c1 < c2$. Les facteurs constants peuvent être beaucoup moins significatifs, au niveau de la durée d'exécution, que la dépendance par rapport au nombre n de données à trier.

Si on compare un ordinateur rapide (ordinateur A) exécutant un tri par insertion et un ordinateur lent (ordinateur B) exécutant un tri par fusion. Ces deux machines doivent chacune trier un tableau d'un million de nombres.

Supposez que l'ordinateur A exécute un milliard d'instructions par seconde et que l'ordinateur B n'exécute que dix millions d'instructions par seconde (A est 100 fois plus rapide que l'ordinateur B)

Supposons que pour le trie de n nombres, l'ordinateur A termine au bout de $2n^2$ instructions et que B termine au bout de $50n \lg n$ instructions.

Alors pour un nombre n = 1 000 000 de nombre entier à trier, l'ordinateur A demande 1jour 9h 20 s

$$\frac{2 \cdot (10^6)^2 \text{ instructions}}{10^9 \text{ instructions/second}} = 2000 \text{ seconds},$$

while computer B takes

$$\frac{50 \cdot 10^6 \lg 10^6 \text{ instructions}}{10^7 \text{ instructions/second}} \approx 100 \text{ seconds}.$$

et que l'ordinateur B demande 1h 40mn.