Introduction à l'algorithmique et à la programmation

Laurent Signac – CC-BY-SA – 25-09-23 1653 11cac5ab28626c41d285

1 Qu'est ce qu'un algorithme ?

Préface à l'édition française de *Introduction à l'algorithmique* de Cormen, Leiserson et Rivest

> *Vous savez compter. Un ordinateur aussi ! Mais connaissez-vous les mécanismes utilisés ? Êtes vous vraiment sûr que le résultat affiché est juste ? Combien de temps devrez-vous attendre la fin du calcul ? N'y a-t-il pas moyen de l'obtenir plus vite ? Que vous soyez ingénieur, mathématicien, physicien, statisticien et surtout informaticien, toutes ces questions, vous vous les posez. Si vous êtes étudiant, elles surgiront très rapidement.*
> *Étudier l'algorithmique, c'est apporter des réponses à vos questions.*

Définition donnée par *Knuth* (*Les algorithme*, chapitre 3) :

> *Un algorithme est un ensemble de règles permettant d'obtenir une sortie spécifique à partir d'une entrée spécifique. Chaque étape doit être définie de façon précise afin d'être traduite dans un langage informatique et exécutée par une machine.*

Knuth ajoute un peu plus loin :

> *On aimerait de plus qu'un algorithme se termine toujours après un nombre fini d'étapes.*

J'ajoute ici qu'on peut considérer l'algorithmique comme la partie *conceptuelle* de la programmation.

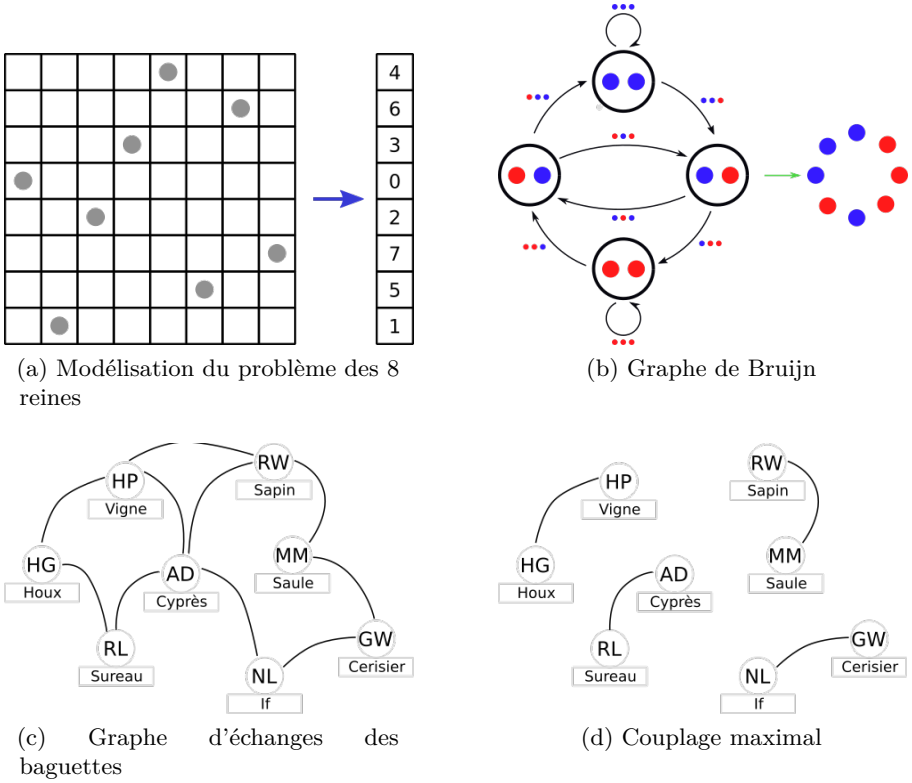
2 Ce qui relève de l'algorithmique

1. **Savoir si un problème particulier peut être résolu ou non avec un algorithme.** Il existe des problèmes pour lesquels il n'y a pas de solution algorithmique (ces problèmes sont dits non-calculables). Par exemple : savoir si un programme arbitraire s'arrête au bout d'un temps fini ; décider si un programme particulier a un comportement viral ; décider si une une équations diophantienne (équation polynomiale à coefficients entiers) a des solutions entières ?
2. **Trouver un algorithme de résolution pour un problème particulier.** La difficulté est très variable : trouver le plus petit élément d'un ensemble, rechercher le flot maximum dans un graphe, écrire un compilateur…
3. **Prouver qu'un algorithme est correct.**
4. **Estimer l'efficacité d'un algorithme.** Comment varie le temps de calcul d'un addition en fonction du nombre de chiffre des opérandes ? Le problème du voyageur de commerce consiste à trouver un circuit passant par n villes qui soit de longueur minimale. Pour combien de villes peut-on le résoudre en pratique ?

Nous seront aussi amenés à **programmer effectivement un algorithme**, en utilisant le langage Python.

Le second point laisse un place importante à la modélisation, qui est parfois négligée. Modéliser correctement le problème (et en déduire les structures de données adaptées) permet de gagner en efficacité.

- le problème des 8 reines consiste à placer 8 reines sur un échiquier de manière à ce qu'il n'y en ait pas deux en prise. Modéliser les positions sur l'échiquier, non pas par un tableau bidimensionnel mais par un tableau contenant la liste des numéros de colonnes (fig. 1a) donne une solution plus élégante.
- une séquence de Bruijn d'ordre n sur k symboles est un mot circulaire contenant toutes les suites de longueur n formées sur les k symboles. Le problème est équivalent à la recherche d'un circuit eulérien (ou hamiltonien) dans un graphe particulier (fig. 1b)
- chaque sorcier a sa baguette, mais envisage un échange en accord avec un autre sorcier (car les deux y trouvent avantage) ; peut-on satisfaire tout le monde ? Le problème consiste à rechercher des couplages dans un graphe (fig. 1c et fig. 1d)



(a) Modélisation du problème des 8 reines

(b) Graphe de Bruijn

(c) Graphe d'échanges des baguettes

(d) Couplage maximal

Figure 1: Modélisation de problème

3 De quoi sont fait les algorithmes ?

Voici l'algorithme de l'exponentiation modulaire tel qu'il apparaît dans *Introduction à l'algorithmique* :

```
EXPONENTIATION-MODULAIRE(a, b, n)
1  c ← 0
2  d ← 1
3  soit ⟨b_k, b_{k−1}, ..., b_0⟩ la représentation binaire de b
4  pour i ← k decr jusqu'à 0
5      faire c ← 2c
6           d ← (d·d) mod n
7           si b_i = 1
8             alors c ← c + 1
9                  d ← (d·a) mod n
10 retourner d
```

Cet algorithme contient une fonction, des instruction de contrôle de flux (boucle, test), des calculs : ce sont les éléments de base des algorithmes, et aussi des langages de programmation impératifs.

Les mêmes éléments reviennent toujours :

- Les **variables**, pour stocker/nommer et manipuler des grandeurs et des objets (d'un certain **type**) (par exemple `a` et `b` dans le calcul du pgdc)
- L'**affectation**
- La **séquence d'instructions** qui exécute une suite d'instructions dans l'ordre
- Les **fonctions** et les **procédures**, pour **isoler** certaines parties du programme, les **traiter à part** pour pouvoir les vérifier plus facilement et/ou les réutiliser
- Les **conditionnelles**, pour effectuer un bloc d'instructions plutôt qu'un autre en fonction de certains résultats
- Les **boucles**, pour répéter plusieurs fois un bloc d'instructions
- Les **tableaux** et les **listes**, des structures de données très utilisées

Les éléments des algorithmes sont très stables, ils se retrouvent dans tous les langages (ou presque). Arriver à les comprendre, à prendre de la hauteur, permet de passer d'un langage à un autre.

4 Les algorithmes au travail

L'algorithmique, Claire Mathieu :

> *Adopter une perspective algorithmique, cela revient à regarder les problèmes en ayant toujours conscience de la dimension de calcul, et de la plus ou moins grande aisance avec laquelle ces objets peuvent être manipulés. Les représentations que l'on choisira, les modélisations que l'on définira, les objectifs que l'on se donnera, les contraintes que l'on s'imposera, tout sera conçu en restant toujours conscient des impacts de nos choix sur les algorithmes.*

Dans le cadre du quotidien professionnel (d'une personne qui n'est pas professionnelle de l'informatique, et en particulier de la programmation), être capable d'automatiser des tâches permet de gagner du temps. La programmation devient alors un moyen pour obtenir une réponse ou un travail effectif. Le soin qu'on apportera à la programmation d'outils permettra de réutiliser le travail effectué une fois pour en tirer un plus grand bénéfice.

5 Les algorithmes dans la société

Avoir une pensée algorithmique permet de mieux comprendre certains aspects de notre société, d'être éventuellement plus méfiant, voire de participer aux décisions qui font l'avenir.

5.1 ParcourSup

L'algorithme de ParcourSup, qui permet d'apparier des étudiants et des études supérieures est le successeur de APB (Admission Post Bac). Les deux algorithmes sont des algorithmes de résolution d'un problème appelé : *problème des mariages stables* et dont une solution a été donnée par Gale et Shapley en 1962. La différence entre les deux vient de ce qu'ABP nécessite que les étudiants classent leurs choix. Ce classement étant accessible aux Universités, une stratégie qui consiste à ne pas être sincère sur ses choix de formation a émergé. Or l'algorithme de Gale Shapley donne une solution optimale si les classements sont sincères uniquement. Le changement pour ParcourSup interdit le classement côté étudiant. Le prix à payer est que l'affectation n'est plus complètement automatique, et nécessite que le candidat intervienne pour valider ses choix (ce qui était fait *a priori* avec le classement APB).

5.2 Découpage électoral

Un district dispose de 4 sièges à pourvoir par une élection. Lors de cette élection (31 votes), on relève 15 voix pour les Orange, et 16 voix pour les Violets. Il y aura donc 2 ou 3 sièges pour les Violets.

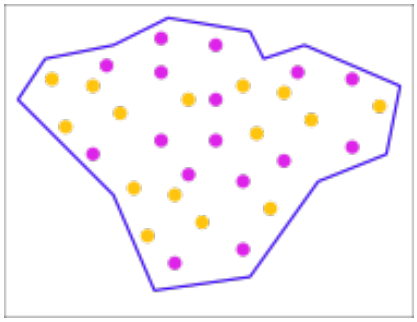


Figure 2: Avec un seul district, les Violets sont majoritaires

On découpe en 4 districts, avec un siège par district. Un des districts aura 1 siège Violet. Les 3 autres auront un siège Orange… la tendance est inversée.

Aux États Unis, le découpage en districts a lieu après chaque recensement selon un algorithme particulier. En 2012, 50,5 % des voies de la Caroline du Nord allaient aux démocrates. Suite au découpage en 13 districts, les démocrates ont obtenu 4 sièges sur les 13 de la Caroline du Nord.

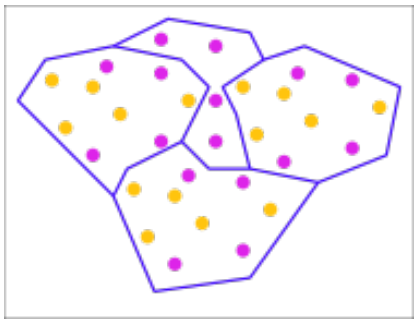


Figure 3: Après découpage en 4 districts, les Violets sont minoritaires

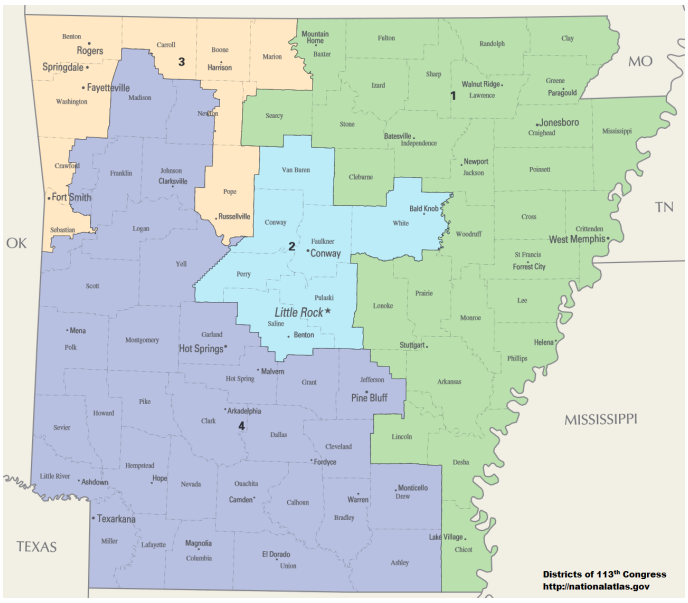


Figure 4: Exemple de découpage de l'Alabama (wikimedia)

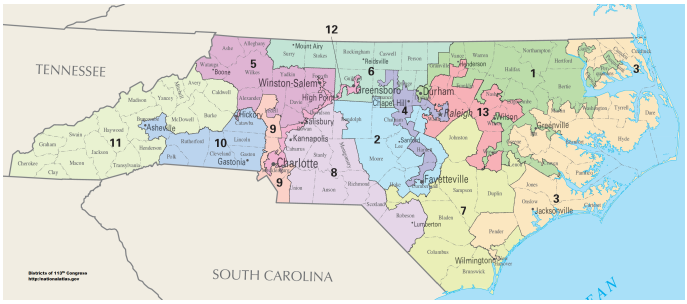


Figure 5: Découpage de la Caroline du Nord (wikimedia)

6 Programmation

6.1 Qu'est-ce qu'un programme ?

Un programme est un texte qui exprime un algorithme transformant de l'information. Il est écrit dans un langage de programmation particulier afin d'être **exécuté en pratique** par une machine (après transformation).

D'après don E. Knuth :

> *Science is what we understand well enough to explain to a computer. Art is everything else we do.*

6.2 Différents types de langage

Il existe plusieurs centaines de langages ;

- certains sont spécialisés, d'autres généralistes ;
- ils permettent d'utiliser certains paradigmes plutôt que d'autres ;
- ils peuvent être verbeux ou expressifs ;
- …

Programmer dans un langage donné, une fois les éléments d'algorithmique connus, est essentiellement une affaire de pratique.