Introduction à l'informatique
Algorithmes

Benjamin Monmege

2019/2020

L'informatique consiste ensuite à appliquer trois piliers (calculabilité, complexité temporelle, complexité spatiale) à la résolution de *tâches complexes*. On se base sur l'*abstraction* pour oublier des détails peu importants. On *décompose* alors le problème en tâches plus simples. Puis on essaie de résoudre ces tâches simples et ciblées avec des *algorithmes*. Une illustration naïve de ces trois temps peut être donnée par une image de brins de laine emmêlés (cf Figure 1). Avant d'avoir tricoté le pull tricolore pour le petit dernier, il faut d'abord démêler les brins pour ne considérer que des pelotes de laine de différentes couleurs : c'est la phase d'abstraction. On décompose ensuite la tâche complexe de tricot d'un pull bicolore en tâche plus simples : il faudra ainsi tricoter une manche bleue, un corps gris et une manche rouge, avant de coudre le tout ensemble.

Des tâches complexes plus réalistes, en terme d'informatique, sont décrites en Figure 2 et 3. On ajoute à la phase d'abstraction et à l'algorithme à proprement parler une phase de visualisation qui est le résultat observé par l'utilisateur : un plus court chemin du campus St Charles au Vieux Port, ou le tri des restaurants de Marseille par leur note moyenne décroissante.

Il nous reste donc à savoir comment résoudre les tâches simples qui proviendront de la décomposition précédente. Résoudre, ou plutôt *faire résoudre* automatiquement par une machine. La description d'une telle méthode de résolution est appelée un *algorithme*. L'étymologie de ce mot provient d'al-Khuwarizmi (cf Figure 4) un savant ayant vécu au début du IXème siècle à Bagdad. Son ouvrage *Abrégé du calcul par la restauration et la comparaison* est à l'origine de l'algèbre : il y décrit et classifie des algorithmes tels que celui d'Euclide pour

Figure 1 – Tricot d'un pull tricolore à partir de brins de laine emmêlés

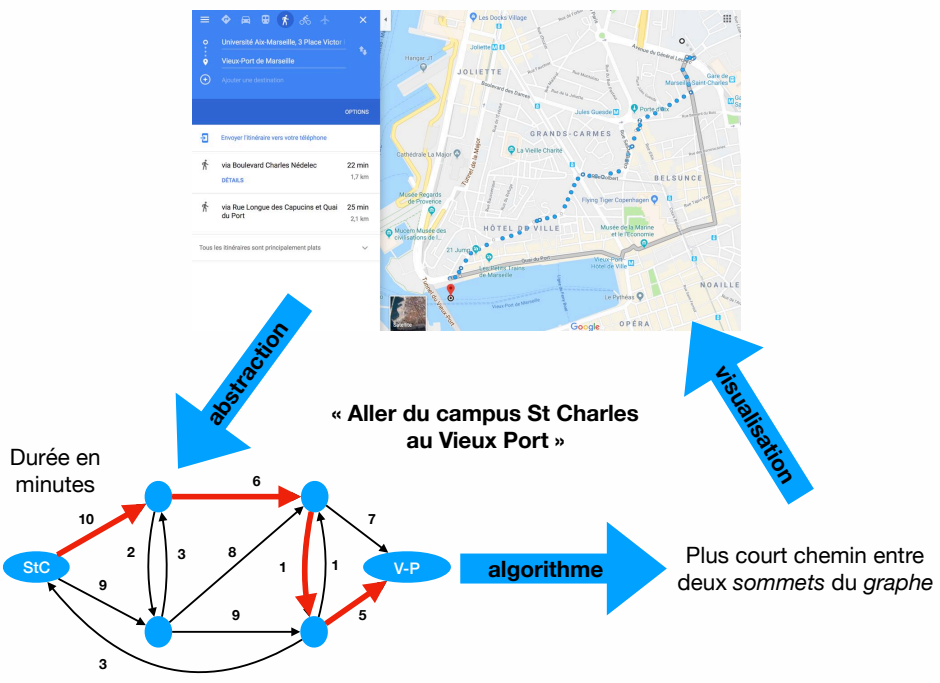

FIGURE 2 – Calcul du plus court chemin dans Google Maps : on abstrait le problème à l'aide d'un graphe avec des nœuds représentant les intersections de rue et des arcs reliant ces nœuds avec la durée en minutes du trajet correspondant

FIGURE 3 – Tri des restaurants par note moyenne décroissante : la seule information pertinente à conserver ici est le nom du restaurant et sa note moyenne. La visualisation consiste alors à réordonner effectivement les restaurants une fois qu'ils ont été triés

al-Khuwarizmi

أبو عبد الله محمد بن موسى الخوارزمي

FIGURE 4 – Le savant al-Khuwarizmi, premier auteur d'algorithmes

le calcul du plus grand commun diviseur de deux entiers ou pour la résolution d'équations du second degré.

Proposons alors une définition de ce qu'est un algorithme :

> « Un algorithme est la description *non ambiguë* d'une séquence *finie* d'instructions permettant de résoudre un problème (informatique) ou d'obtenir un résultat. »

Cinq conditions (ainsi décrites par Donald Knuth, un chercheur contemporain en informatique) sont à réunir pour être sûr qu'on a bien un algorithme :

1. **finitude** : un algorithme doit terminer après un nombre fini d'étapes ;
2. **définition précise** : un algorithme doit être décrit sans ambiguïté, c'est-à-dire sans laisser de place au doute ;
3. **entrées** : un algorithme peut avoir des données sur lesquelles il va travailler ;
4. **sorties** : un algorithme a généralement un résultat qu'on attend de lui ;
5. **rendement** : un algorithme doit utiliser des opérations basiques, généralement telles qu'un être humain puisse les exécuter (il n'est donc pas autorisé d'utiliser une opération qui consisterait à construire un carré de même aire qu'un cercle donné, à l'aide d'une règle et d'un compas !).

Illustrons le concept d'algorithme à l'aide d'un problème concret, celui de compter le nombre de personnes dans une salle (par exemple une salle de concert ou un amphithéâtre à l'université). Il existe de multiples méthodes pour opérer ce dénombrement. Le plus simple est qu'une personne compte un par un les personnes de la salle. Il peut aussi choisir de compter les personnes deux par deux, ou même cinq par cinq. Le nombre d'*opérations élémentaires* qu'il effectue diffère dans ces trois cas :

— lorsqu'il compte un par un, il effectue autant d'additions qu'il y a de personnes dans la salle, disons n personnes ;
— lorsqu'il compte deux par deux, il effectue $n/2$ additions ;

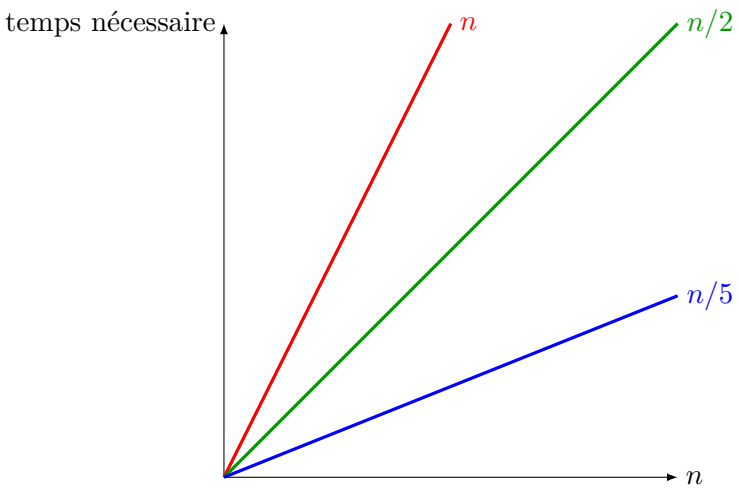

FIGURE 5 – Nombre d'opérations des différents algorithmes en fonction du nombre de personnes n dans la salle

— lorsqu'il compte cinq par cinq, il effectue $n/5$ additions.

Au prix de savoir faire des additions deux par deux ou cinq par cinq rapidement, on voit bien qu'on effectue moins de calculs en comptant cinq par cinq qu'en comptant deux par deux, et que cette dernière technique effectue moins de calculs que celle qui consiste à compter un par un. Une représentation du nombre d'opérations en fonction de n peut être vue en FIGURE 5.

Une autre méthode de dénombrement consiste, si les personnes sont assises, à compter le nombre R de rangées de sièges et à estimer le nombre M moyen de personnes par rangée : il ne reste alors plus qu'à effectuer la multiplication $R \times M$ pour obtenir une estimation du nombre de personnes. Contrairement aux méthodes précédentes, celle-ci n'apporte qu'une réponse approximative : elle n'est pas (absolument) correcte. Par contre, elle est bien plus rapide encore.

Dans les méthodes précédentes, plutôt que de tout faire tout seul, la personne en charge du comptage pourrait se faire aider : plus on est nombreux à travailler, moins de temps la tâche prendra. En informatique, c'est l'utilisation des multiples *cœurs* d'un processeur que nous pouvons utiliser en parallèle pour accélérer la résolution d'un problème.

Une autre façon de paralléliser, beaucoup plus massive, consiste à distribuer le calcul sur les personnes présentes dans la salle. Considérons ainsi l'algorithme suivant :

Entrée : des personnes debout dans une salle
- Chaque personne a en tête le nombre 1
- **Tant qu**'il reste au moins deux personnes debout :
  - chaque personne encore debout cherche du regard une autre personne debout
  - les deux personnes s'échangent le nombre qu'ils ont en tête (indépendamment des autres personnes)
  - l'une des deux personnes s'assoit ; l'autre additionne les deux nombres et reste debout

Sortie : la dernière personne debout crie son nombre

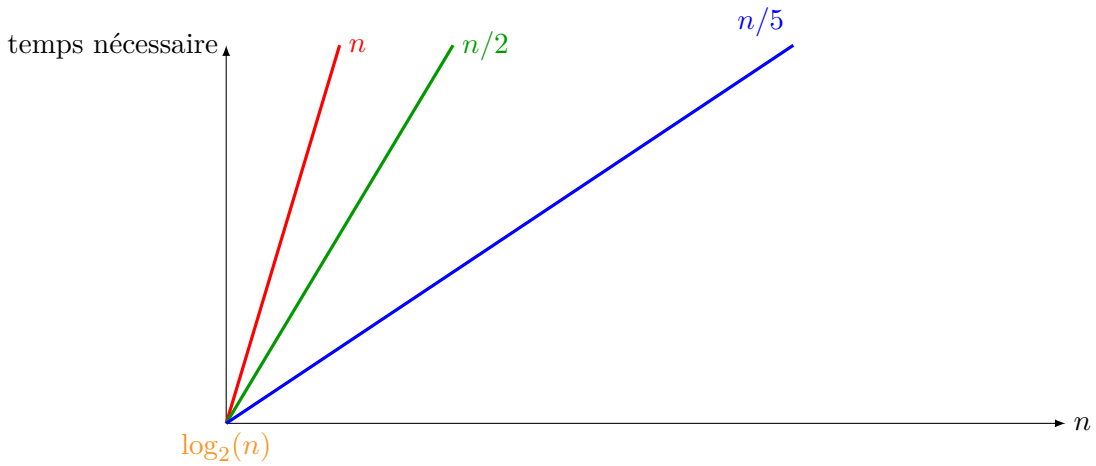

FIGURE 6 – Comparaison des fonctions linéaires (en rouge, vert et bleu) avec la fonction logarithme (en orange)

Cet algorithme est correct au sens où il se termine avec une seule personne encore debout, et que le nombre crié est effectivement le nombre de personnes dans la salle. Combien de temps prend-il pour terminer ? Cette fois-ci, plutôt que le nombre total d'additions effectuées, une meilleure estimation du temps d'exécution consiste à compter le nombre d'*itérations* effectuées par l'algorithme. Dis autrement, si on suppose que toutes les secondes, chaque personne encore debout en a trouvé une autre et que l'une des deux s'est assise, alors il s'agit de compter le nombre de secondes avant la fin de l'algorithme. Il n'est pas très difficile de se convaincre qu'à chaque seconde, la moitié environ des personnes encore debout s'assoit. Par conséquent, après k secondes, le nombre de personnes encore debout a été divisé par 2^k. Lorsque k devient supérieur à $\log_2(n)$, le nombre de personnes debout a été divisé par $2^{\log_2(n)} = n$: il ne reste alors plus qu'une seule personne encore debout. Ainsi, il faut $\log_2(n)$ secondes pour que cet algorithme termine. La fonction logarithme croit beaucoup plus lentement que les fonctions linéaires trouvées précédemment. Dans la FIGURE 6, on peut observer que lorsque n est suffisamment grand, la courbe de la fonction $\log_2$ passe en dessous des courbes des fonctions $n \mapsto n$, $n \mapsto n/2$ et $n \mapsto n/5$. En particulier, s'il y a 200 personnes dans la salle et que chaque opération élémentaire (addition ou itération) nécessite une seconde :

- la méthode consistant à compter un par un nécessite 200 secondes, soit 3 minutes et 20 secondes ;
- la méthode consistant à compter deux par deux nécessite 100 secondes, soit 1 minutes et 40 secondes ;
- la méthode consistant à compter cinq par cinq nécessite 40 secondes ;
- la méthode distribuée nécessite $\log_2(200)$ secondes, soit 7 secondes environ : imbattable !