Algorithmie : historique

Julien Noyer

2018-09-20

Algorithmie *Présentation générale*

Un algorithme est une suite d'instructions (ou commandes) qui permettent d'aboutir à un ou plusieurs résultats. A la manière d'une recette de cuisine, un algorithme prend en compte des informations entrantes (ingrédients, ustensiles, temps de cuisson, …) dans un contexte défini (cuisine familiale, fast-food, grand restaurant, …) pour obtenir un résultat attendu par celui qui l'exécute (quiche lorraine, hamburger, blanc mangé, …). Comme en cuisine il existe un nombre incalculable d'algorithmes différents pour arriver au résultat attendu, la question n'est donc pas de trouver l'algorithme absolu qui puisse résoudre tous les opération mais de définir son cadre d'exécution afin de définir les étapes les plus optimisées pour réduire l'impact de son exécution. Pour reprendre le parallèle avec la cuisine, le chef d'un grand restaurant est capable de réaliser un hamburger d'exception mais si vous n'avez que 20 minutes pour manger il est préférable d'aller dans un Fast-food : le but de l'algorithme n'est donc pas de réaliser un hamburger mais de trouver le meilleur moyen pour manger un hamburger rapidement.

Hamburger « Fast-Food » ou hamburger « Prestige » ?

Avoir la charge de la création d'un algorithme nécessite d'avoir la capacité de découpé une interrogation complexe en suite de questions simples afin de définir les étapes de résolution d'un problème. Le but de l'algorithme n'est donc pas de certifier à coup sur que la réponse à l'interrogation sera trouver mais de définir les étapes à suivre pour résoudre le problème identifier dans l'interrogation

De fait, il existe un grand nombre d'algorithme différents pour résoudre les mêmes problèmes, le but est d'estimer lequel choisir selon les contraintes de départ. Nous allons analyser deux algorithmes différents pour identifié celui qui sera la meilleure solution pour répondre à la problématique : *« comment manger un hamburger rapidement »*.

Tableau 1 : réalisation « Fast-Food »

Dans le cadre du *« Fast-Food »* nous considérons que tous les ingrédients sont pré-fabriqués et accessibles sur le plan de travail du cuisinier.

Eléments	Actions	Temps (s)
Pain	Sortir de l'emballage	10
	Découper en deux	10
Salade	Découper une feuille	10
	Laver la feuille	10
Tomate	Sortir de l'emballage	10
	Laver et découper	40
Oignon	Sortir de l'emballage	10
	Eplucher et découper	45
Steak	Sortir de l'emballage	10
	Cuire	180
Préparation	Assembler les ingrédients	20
	Emballer le burger	10

Tableau 2 : réalisation « Prestige »

Dans le cadre du *« Prestige »* nous considérons que tous les ingrédients sont fait par le cuisinier ou accessible depuis le potager associé à la cuisine.

Eléments	Actions	Temps (s)
Pain	Préparer la pâte	10
	Laisser fermenter	10
	Façonner	10
	Cuire	10
	Découper en deux	10
Salade	Cueillir dans le potager	10
	Découper une feuille	10
	Laver une feuille	10
Tomate	Cueillir dans le potager	40
	Laver et découper	40
Oignon	Cueillir dans le potager	45
	Eplucher et découper	45
Steak	Découper des morceaux de boeuf	180
	Hacher	180
	façonner	180
	Cuire	180
Préparation	Décorer une assiette	10
	Assembler les ingrédients	10
	Placer le burger sur une assiette	10

Plus il y a de contraintes plus c'est facile

La démonstration ci-dessus est flagrante : il faut 365 secondes pour réaliser un hamburger « Fast-Food » quand il en fait 820 pour le hamburger « Prestige ». Le fait de devoir manger vite nous permet facilement de définir l'algorithme le plus efficace mais qu'adviendrait-il si nous changeons le nombre de cuisinier

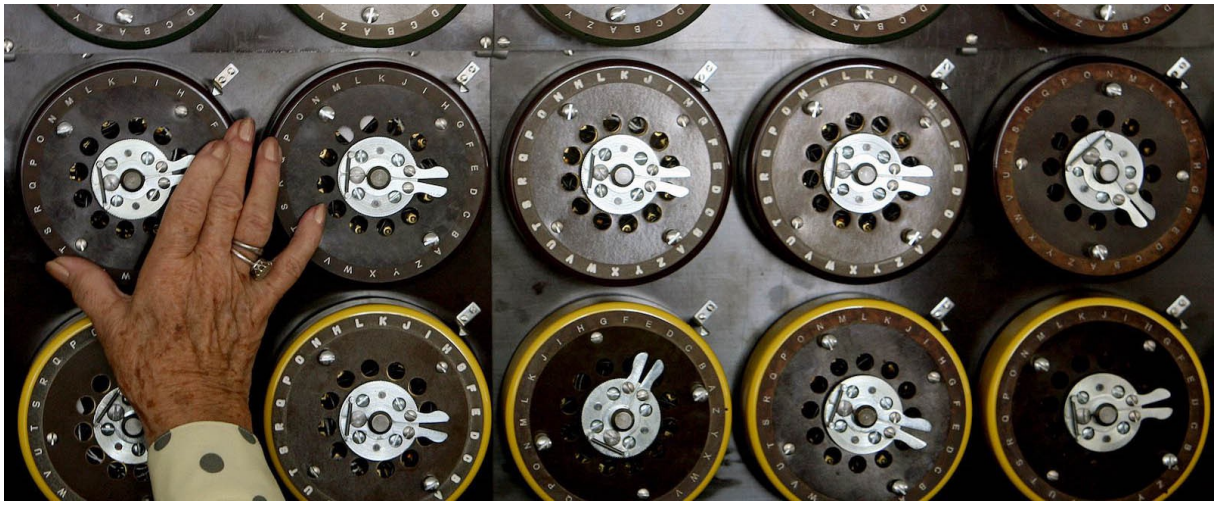
dans l'algorithme « Prestige » ?

Au-delà du nombre de cuisinier, quelles sont le moyen que nous pouvons mettre en œuvre pour accélérer l'exécution des algorithmes ?

— Les actions doivent-elles être faite les unes après les autres ?
— Est-il possible de changer l'ordre des étapes ?
— Les hamburger sont-ils réalisés à la demande ?
— Qu'est-il possible de préparer en avance ?
— Est-il possible de modifier les quantités, les cuissons ?

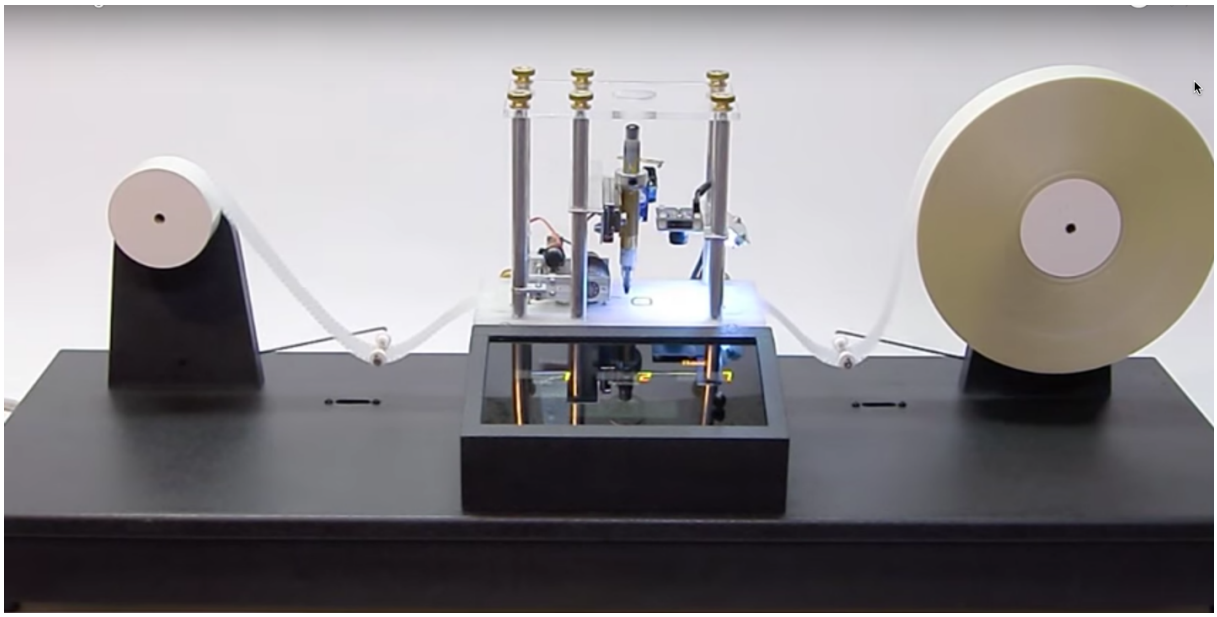
Du cuisinier au Robot Cuiseur

Les étapes que nous avons présentées dans les tableaux 1 et 2 peuvent permettre à un humain de réaliser le hamburger souhaité - avec plus ou moins de dextérité - mais quand est-il si nous souhaitons faire réaliser ces opérations par une machine ? Nous aurions certainement besoin de passer par un langage informatique afin de faire comprendre à la machine les instructions que nous aurons défini dans notre algorithme ce qui nécessiterait des connaissances spécifiques inhérentes à l'utilisation de la machine. Mais si un algorithme est capable de définir le processus pour réaliser une recette de cuisine il est également pour définir celui d'une machine, d'un ordinateur.



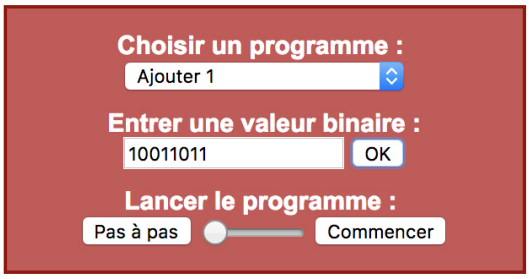
En 1936 Alan Turing à développé un model permettant de définir le fonctionnement d'appareils mécanique qui à ensuite été appelé « Machine de Turing ». Le principe est de définir les étapes qu'un programme doit suivre pour permettre aux appareils de fonctionner, soit un algorithme permettant de définir le fonctionnement de machine qui à l'époque n'existaient pas encore. Le principe de cette machine est de représenter une personne virtuelle exécutant une procédure bien définie qui permettent de changer le contenu d'un tableau par des symboles spécifique.

La machine imaginée par Turing comporte un ruban divisé en cases, dans lesquelles elle peut écrire des symboles. La machine ne peut lire qu'une seule case à la fois, de même elle écrit dans une seule case et décale le ruban d'une seule case vers la gauche ou vers la droite. Les symboles sont en nombre fini. Pour que sa machine fonctionne comme une machine à calculer en binaire, Turing envisage le cas particulier où les symboles utilisés sont 0 et 1. C'est une telle machine que nous vous proposons de tester sur l'animation interactive suivante. Vous pourrez y suivre l'exécution de six « programmes ».



Ajouter 1 à un nombre binaire

Nous allons tester une des principales actions que Alan Turing à voulu faire réaliser par une machine : ajouter 1 à un nombre binaire.



Etat	Lit	Ecrit	Déplace	Suivant
e1	VIDE	VIDE	gauche	e2
e2	0	0	gauche	e2
	1	1	gauche	e2
	VIDE	VIDE	droite	e3
e3	0	1	droite	fin
	1	0	droite	e3
	VIDE	1	droite	fin

Etat de départ : e1

> Cliquer sur l'image pour accéder au simulateur

Tester le simulateur vous permet de comprendre la suite logique de l'algorithme défini par Alan Turing, les étapes à suivre et de valider la stabilité de la suite logique qu'il contient. Il faut néanmoins préciser certains points pour permettre de comprendre le résultat de l'addition.

Pour ajouter 1 à un nombre binaire, nous utilisons la table d'addition suivante :

Table d'addition

Addition	Résultat
0 + 0	0
0 + 1	1
1 + 0	1
1 + 1	10 (on pose 0, et on retient 1)

Cette table d'addition nous permet d'effectuer de tête des additions en binaire simples mais pour des calcules plus complexes il faut suivre une logique de lecture gauche/droite en respectant le principe d'inversion des valeur de 1 à 0 à 1.

Chiffre se termine par 0 : 1001010 + 1 = 1001011

On parcourt le nombre de gauche à droite, et lorsqu'on arrive à la fin, si on trouve un 0, on le remplace par un 1 et on passe à l'état final.

Chiffre se termine par 1 : 1001011 + 1 = 1001100

Après avoir parcouru le nombre de gauche à droite, on revient de droite à gauche, tant qu'on trouve un 1, on le remplace par un 0, lorsqu'on trouve la première occurrence de 0, on la remplace par 1 puis on passe à l'état final.

Chiffre uniquement composé de 1 : 1111111 + 1 = 10000000

Après avoir parcouru le nombre de gauche à droite, on revient de droite à gauche, tant qu'on trouve un 1, on le remplace par un 0 et lorsqu'on arrive au dernier 1, on insère un 1 dans la case vide à gauche et

on passe à l'état final.

La table d'addition et les calcules ci-dessus n'ont pas pour simple but d'explorer l'arithmétique binaire - ou plus simplement le calcul binaire - mais ils permettent de faire le lien avec la ou les machines qui exécuterons nos algorithmes. Car les deux chiffres 0 et 1 sont traduit par la machine en la tension ou passage d'un courant électrique. Par exemple, le 0 peut être représenté par l'état éteint (tension ou courant nul) et 1 par l'état allumé (tension qui existe ou courant qui passe).

De la logique à l'action

Une fois la logique comprise, comment pouvons nous mettre en place une suite d'actions à réaliser pour qu'une machine soit capable d'effectuer nos calcules binaires ? En suivant la logique de la machine de Turing, nous devons envoyer une suite binaire dans une machine et lui demander d'y ajouter 1, l'algorithme installé sur la machine va alors lire chacun des chiffres de gauche à droite, les réécrire et selon le dernier chiffre lu, passer à celui de droite ou revenir vers la gauche pour finalement terminer l'exécution de l'algorithme lorsque le résultat est trouvé.

Table d'action pour ajouter 1 à un nombre binaire

La machine de Turing utilise donc pour notre calcule la table d'actions suivante :

Etat	Lit	Ecrit	Déplace	Suivant
e1	VIDE	VIDE	gauche	e2
e2	0	0	gauche	e2
	1	1	gauche	e2
	VIDE	VIDE	droite	e3
e3	0	1	droite	fin
	1	0	droite	e3
	VIDE	1	droite	fin

Quel que soit le nombre envoyé dans la machine, cette dernière utilise la table d'action et agit de la manière suivante :

- Si elle est à l'état 1 et trouve une case vide :
  - elle n'écrit rien
  - décale sa tête de lecture d'une case à gauche

    - passe à l'état 2
- Si elle est à l'état 2 et lit un 0 :
    - elle écrit 0
    - décale sa tête de lecture d'une case à gauche
    - reste à l'état 2
- Si elle est à l'état 2 et lit un 1 :
    - elle écrit 1
    - décale sa tête de lecture d'une case à gauche
    - reste à l'état 2
- Si elle est à l'état 2 et trouve une case vide :
    - elle n'écrit rien
    - décale sa tête de lecture d'une case à droite
    - passe à l'état 3
- Si elle est à l'état 3 et lit un 1 :
    - elle écrit 0
    - décale sa tête de lecture d'une case à droite
    - reste à l'état 3
- Si elle est à l'état 3 et lit un 0 :
    - elle écrit 1
    - décale sa tête de lecture d'une case à droite
    - passe à l'état final
- Si elle est à l'état 3 et trouve une case vide :
    - elle écrit 1
    - décale sa tête de lecture d'une case à droite
    - passe à l'état final