Benshi
Ciné Club

The Imitation Game

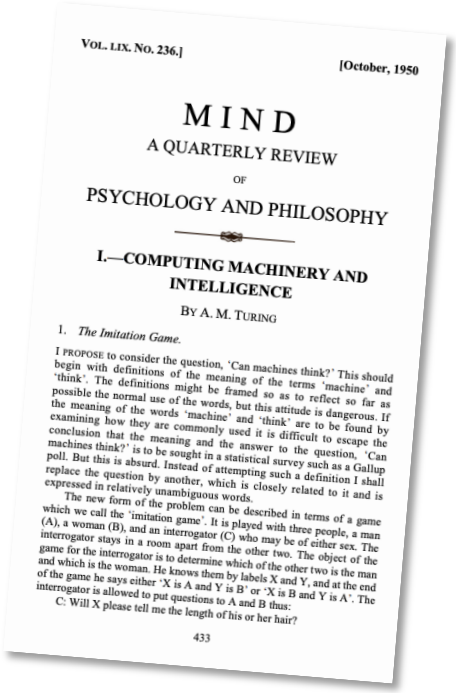
CINEMATHEMATIQUES

DOSSIER PEDAGOGIQUE

LYCEE

Le film

Imitation Game est un film biographique américain réalisé par Morten Tyldum, sorti en 2014. Il s'agit de l'adaptation cinématographique de la biographie *Alan Turing ou l'énigme de l'intelligence* d'Andrew Hodges. Le film est inspiré de la vie du mathématicien et cryptanalyste britannique Alan Turing, notamment pendant la Seconde Guerre mondiale, lorsqu'il a travaillé à déchiffrer le code secret nazi : Enigma.



VOL. LIX. NO. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

BY A. M. TURING

1. *The Imitation Game.*

I PROPOSE to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, 'Can machines think?' is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words.

The new form of the problem can be described in terms of a game which we call the 'imitation game'. It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either 'X is A and Y is B' or 'X is B and Y is A'. The interrogator is allowed to put questions to A and B thus:

C: Will X please tell me the length of his or her hair?

433

Le titre du film est une référence à l'introduction de l'article écrit par Alan Turing en 1951 pour présenter ses recherches sur l'intelligence artificielle et notamment ce qui est devenu par la suite le test de Turing. Celui-ci est brièvement évoqué dans le film, mais n'est pas le sujet principal du film, qui traite essentiellement de son travail sur Enigma et de son homosexualité.

Le contexte historique

En 1940, les Britanniques sont en train de perdre la guerre à cause des messages secrets nazis qu'ils ne réussissent pas à décrypter. Le gouvernement décide alors d'employer de nouveaux mathématiciens pour tenter de briser ce code. Alan Turing fait partie des nouvelles recrues et il deviendra vite le chef des opérations.

Pour décrypter les messages plus rapidement, Alan Turing va imaginer une machine capable de décrypter automatiquement les messages. Une idée révolutionnaire pour l'époque qui permettra de décrypter les messages d'Enigma. Selon les historiens, le travail d'Alan Turing et de ses équipes a permis d'écourter la guerre d'1 à 2 années et de sauver la vie de 14 millions de personnes. Le héros de la seconde guerre mondiale est donc un mathématicien.

Mettre en scène une recherche mathématique

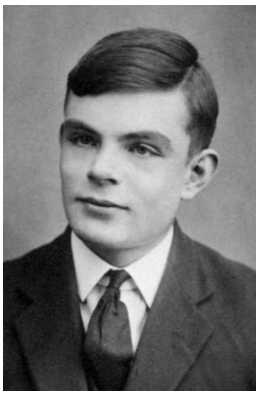
Le biopic est un genre souvent complexe à aborder car il doit présenter la complexité d'un personnage sans tomber dans l'hagiographie béate. Afin d'éviter cet écueil *The Imitation*

Game va s'appuyer sur la figure quasi antihéroïque d'Alan Turing dont les traits de caractères vont être renforcés à des fins purement fictionnelles, le faisant osciller entre enfermement autistique et misanthropie. En parallèle, le cinéaste va tisser un récit inversé puisqu'il commence par la fin, ce qui lui permet de donner d'emblée un ton résolument dramatique. Ce renversement va permettre au cinéaste d'utiliser les flashes-backs comme les moteurs dramatiques principaux du film.

Ce procédé narratif est d'autant plus limpide qu'il sert un suspense absolument central dans ce récit historique. L'agencement des flashes-back fonctionne comme une suite de tableaux qui cultive habillement le mystère sur le destin d'Alan Turing. Le rythme du film est ainsi tendu par la course contre la montre en jeu puisque que du succès de l'entreprise dépend la fin de la guerre. La métaphore est d'ailleurs filée par le cinéaste qui insiste sur la répétition des plans de courses à pied du mathématicien ou encore avec un montage parallèle sur les roulements de la machine de Turing et les chenilles des chars nazis.

Cette temporalité éclatée du récit permet également de lire le présent par le passé, tout en rythmant le parcours du personnage. L'enjeu n'est plus tant la résolution du code que de comprendre la psychologie des personnages et les moyens mis en œuvre pour y parvenir. De plus, le protagoniste prend une nouvelle épaisseur humaine et devient un génie incompris et même au regard de notre propre époque un génie sacrifié. Les héros d'un moment étant injustement oubliés et même honnis au nom d'enjeux militaires et de l'homosexualité de Turing. Cet entremêlement d'époque devient alors un outil de mise en perspective politique du passé.

Le mathématicien



Alan Mathison Turing, né le 23 juin 1912 à Londres et mort le 7 juin 1954 à Wilmslow, est un mathématicien et cryptologue britannique, auteur de travaux qui fondent scientifiquement l'informatique.

Pour résoudre le problème fondamental de la décidabilité en arithmétique, il présente en 1936 une expérience de pensée que l'on nommera ensuite machine de Turing et des concepts de programme et de programmation, qui prendront tout leur sens avec la diffusion des ordinateurs, dans la seconde moitié du XX^{e} siècle. Son modèle a contribué à établir la thèse de Church, qui définit le concept mathématique intuitif de fonction calculable.

Durant la Seconde Guerre mondiale, il joue un rôle majeur dans la cryptanalyse de la machine Enigma utilisée par les armées allemandes : l'invention de machines usant de procédés électroniques, les bombes, fera passer le décryptage à plusieurs milliers de messages par jour. Ce travail secret ne sera connu du public que dans les années 1970. Après la guerre, il travaille sur un des tout premiers ordinateurs, puis contribue au débat sur la possibilité de l'intelligence artificielle, en proposant le test de Turing. Vers la fin de sa vie, il s'intéresse à des modèles de morphogenèse du vivant conduisant aux « structures de Turing ».

Poursuivi en justice en 1952 pour homosexualité, il choisit, pour éviter la prison, la castration chimique par prise d'œstrogènes. Il est retrouvé mort par empoisonnement au cyanure le 8 juin 1954 dans la chambre de sa maison à Wilmslow. La reine Élisabeth II le reconnaît comme héros de guerre et le gracie à titre posthume en 2013.

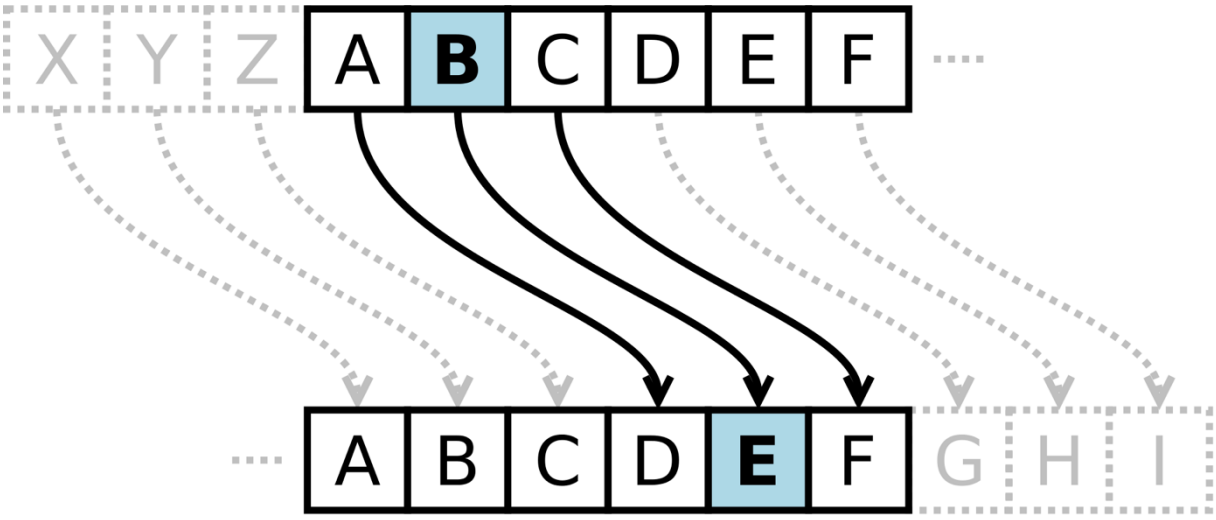
Un exemple de cryptage : le code César

En cryptographie, le chiffrement par décalage, aussi connu comme le chiffre de César ou le code de César (voir les différents noms), est une méthode de chiffrement très simple utilisée par Jules César dans ses correspondances secrètes (ce qui explique le nom « chiffre de César »).

Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe, toujours du même côté, dans l'ordre de l'alphabet. Pour les dernières lettres (dans le cas d'un décalage à droite), on reprend au début. Par exemple avec un décalage de 3 vers la droite, A est remplacé par D, B devient E, et ainsi jusqu'à W qui devient Z, puis X devient A etc. Il s'agit d'une permutation circulaire de l'alphabet. La longueur du décalage, 3 dans l'exemple évoqué, constitue la clé du chiffrement qu'il suffit de transmettre au destinataire — s'il sait déjà qu'il s'agit d'un chiffrement de César — pour que celui-ci puisse déchiffrer le message. Dans le cas de l'alphabet latin, le chiffre de César n'a que 26 clés possibles (y compris la clé nulle, qui ne modifie pas le texte).

Il s'agit d'un cas particulier de chiffrement par substitution monoalphabétique : ces substitutions reposent sur un principe analogue, mais sont obtenues par des permutations quelconques des lettres de l'alphabet. Dans le cas général, la clé est donnée par la permutation, et le nombre de clés possibles est alors sans commune mesure avec celui des chiffrements de César.

Le chiffrement de César a pu être utilisé comme élément d'une méthode plus complexe, comme le chiffre de Vigenère. Seul, il n'offre aucune sécurité de communication, à cause du très faible nombre de clés, ce qui permet d'essayer systématiquement celles-ci quand la méthode de chiffrement est connue, mais aussi parce que, comme tout encodage par substitution monoalphabétique, il peut être très rapidement « cassé » par analyse de fréquences (certaines lettres apparaissent beaucoup plus souvent que les autres dans une langue naturelle).



Le code César avec Python

Manipuler les variables « string » avec Python

Les variables « string » sont des variables alphanumériques (variables qui peuvent contenir des chaines de caractères).
Voici quelques fonctions applicables à ce type de variable.
Soit v, une variable contenant la chaîne « ABCD » :

v.lower()
Converti tous les caractères de la chaine v en minuscule.

Len(v)
Renvoie le nombre de caractères dans la chaine v.

v[n]
Renvoie la valeur du n^{ième} caractères de la chaine v.

ord(" a ")
Renvoie le code ASCII d'un caractère.
La mémoire de l'ordinateur conserve toutes les données sous forme numérique. Il n'existe pas de méthode pour stocker directement les caractères. Chaque caractère possède donc son équivalent en code numérique : c'est le code ASCII (American Standard Code for Information Interchange - traduisez « Code Americain Standard pour l'Echange d'Informations »). Le code ASCII de base représentait les caractères sur 7 bits (c'est-à-dire 128 caractères possibles, de 0 à 127).

0	*NUL*	16	*DLE*	32	SPC	48	0	64	@	80	P	96	`	112	p
1	*SOH*	17	*DC1*	33	!	49	1	65	A	81	Q	97	a	113	q
2	*STX*	18	*DC2*	34	"	50	2	66	B	82	R	98	b	114	r
3	*ETX*	19	*DC3*	35	#	51	3	67	C	83	S	99	c	115	s
4	*EOT*	20	*DC4*	36	$	52	4	68	D	84	T	100	d	116	t
5	*ENQ*	21	*NAK*	37	%	53	5	69	E	85	U	101	e	117	u
6	*ACK*	22	*SYN*	38	&	54	6	70	F	86	V	102	f	118	v
7	*BEL*	23	*ETB*	39	'	55	7	71	G	87	W	103	g	119	w
8	*BS*	24	*CAN*	40	(	56	8	72	H	88	X	104	h	120	x
9	*HT*	25	*EM*	41	)	57	9	73	I	89	Y	105	i	121	y
10	*LF*	26	*SUB*	42	*	58	:	74	J	90	Z	106	j	122	z
11	*VT*	27	*ESC*	43	+	59	;	75	K	91	[	107	k	123	{
12	*FF*	28	*FS*	44	,	60	<	76	L	92	\	108	l	124	|
13	*CR*	29	*GS*	45	-	61	=	77	M	93	]	109	m	125	}
14	*SO*	30	*RS*	46	.	62	>	78	N	94	^	110	n	126	~
15	*SI*	31	*US*	47	/	63	?	79	O	95	_	111	o	127	DEL

chr(97)
Renvoie le code le caractère associé au code ASCII.

Proposition de codage des fonctions codage/décodage avec Python :

```
##################
# Fonction CODAGE #
##################
def cod(mot,clef):
'''
Cette fonction permet de coder un mot en utilisant le code César.
Le premier argument doit être une chaine de caractères non accentués et entre guillemets. Par exemple
cod("Cesar",2) pour coder le mot César.
Le second argument indique la clef de chiffrement (2 dans l'exemple précèdent).
'''
mot=mot.lower()
mes=""

for i in range(len(mot)):
        if (ord(mot[i])+clef)>122: mes=mes+chr(ord(mot[i])-26+clef)
        else: mes=mes+chr(ord(mot[i])+clef)
return mes


####################
# Fonction DECODAGE #
####################
def decod(mot,clef):
'''
Cette fonction permet de décoder un mot en utilisant le code César.
Le premier argument doit être une chaine de caractères non accentués et entre guillemets. Par exemple
decod("Cesar",2) pour décoder le mot César.
Le second argument indique la clef de chiffrement (2 dans l'exemple précèdent).
'''
mot=mot.lower()
mes=""

for i in range(len(mot)):
        if (ord(mot[i])-clef)<97: mes=mes+chr(ord(mot[i])+26-clef)
        else: mes=mes+chr(ord(mot[i])-clef)
return mes
```

Situation problème :

Le message suivant a été codé à l'aide du code César. Malheureusement, la clef de chiffrement a été perdue...

jzidw xwcz bma kixikqbma lm kzgxbivitqabm

Peux-tu traduire ce message ?

Piste :
Pour trouver la solution, tu peux réaliser un programme qui teste les 26 clefs de décryptage possible...

Proposition de programme :

césar.py

```
01| ##########
02| # CODAGE #
03| ##########
04|
05| def cod(mot,clef):
06|     '''
07|     Cette fonction permet de coder un mot en utilisant le code César.
08|
09|     Le premier argumet doit être une chaine de caractères non accentuées et entre
guillemets.
10|
11|     Par exemple cod("Cesar",2) pour coder le mot César.
12|
13|     Le second argument inidque la clé de chiffrement (2 dans l'exemple précédent).
14|
15|
16|     '''
17|     mot=mot.lower()
18|     mes=""
19|     for i in range(len(mot)):
20|         if (ord(mot[i])+clef)>122:
21|             mes=mes+chr(ord(mot[i])-26+clef)
22|         else:
23|             mes=mes+chr(ord(mot[i])+clef)
24|     return mes
25|
26| ############
27| # DECODAGE #
28| ############
29|
30| def decod(mot,clef):
31|     '''
32|     Cette fonction permet de décoder un mot en utilisant le code César.
33|
34|     Le premier argumet doit être une chaine de caractères non accentuées et entre
guillemets.
35|
36|     Par exemple décod("Cesar",2) pour décoder le mot César.
37|
38|     Le second argument inidque la clé de chiffrement (2 dans l'exemple précédent).
39|
40|
41|     '''
42|     mot=mot.lower()
43|     mes=""
44|     for i in range(len(mot)):
45|         if (ord(mot[i])-clef)<97:
46|             mes=mes+chr(ord(mot[i])+26-clef)
47|         else:
48|             mes=mes+chr(ord(mot[i])-clef)
49|     return mes
50|
51| ####################
52| #     PROGRAMME    #
53| ####################
54|
55|
56| message=str(input("Entrez le mot à tester :"))
57| for c in range (26):
58|     print("Clef de chiffrement :",c+1," --> ",decod(message,c+1))
59|
60|
```