le codage binaire

par ???

collège de Villetaneuse, lycées d'Epinay & Saint Denis (93) — ***codage avec contrainte***

Un message se présente comme une suite de "0" et de "1", mais si on envoie plus de trois "1" à la suite, il risque d'y avoir une erreur de lecture … comment coder les messages, et les décoder à la réception ?

Nous allons vous présenter notre travail, appelé « codage avec contrainte ». Notre travail nous a amenés à l'étude du langage binaire. C'est un langage utilisé par les ordinateurs et les machines. Son alphabet est composé de bits, c'est-à-dire de « 0 » et de « 1 ». Il permet de créer des programmes et de transmettre des données. Le but de notre recherche est de trouver un moyen de transformer et de décoder un message [binaire :] écrit en une suite de « 0 » et de « 1 ».

Le problème est le suivant : on a supposé que la suite de quatre « 1 » provoquait des pertes de données. On rencontre ce problème lors du transfert d'un message binaire par ligne téléphonique et lorsqu'on enregistre une disquette.

Une première solution.

Une solution trouvée est de rajouter un « 0 » toutes les trois positions, de façon à ce qu'il ne puisse y avoir quatre « 1 » de suite.

Supposons un message de départ, quelconque :

1 1 1 0 0 1 1 1 0.

Nous le coderons en rajoutant un « 0 » toutes les trois positions :

1 1 1 0 0 0 1 0 1 1 0 0

Puis, nous transmettrons le nouveau message à un téléphone qui transmet le message codé (111000101100) à un second téléphone. Puis, celui-ci décode le message en retirant les « 0 » ajoutés toutes les quatre positions par le codage.

1 1 1 ~~0~~ 0 0 1 ~~0~~ 1 1 0 ~~0~~

[On obtient] donc

1 1 1 0 0 1 1 1 0,

qui est le message initial.

Une seconde solution.

Elle consiste à remplacer certains groupes par des blocs de trois chiffres :

0	par	0
.10.		111
.11.		110
.111.		101
.1111.		100

« . » peut prendre la valeur de 0 et de 1 par exemple. Le message :

0 1 1 1 1 1 0 0 1 1 0 0
0 1111 10 0 11 0 0

donnera

0 100 111 0 110 0 0

Le message ainsi obtenu répond à la contrainte et pourra être ainsi transféré, puis décodé à la réception de celui-ci :

Exemple.

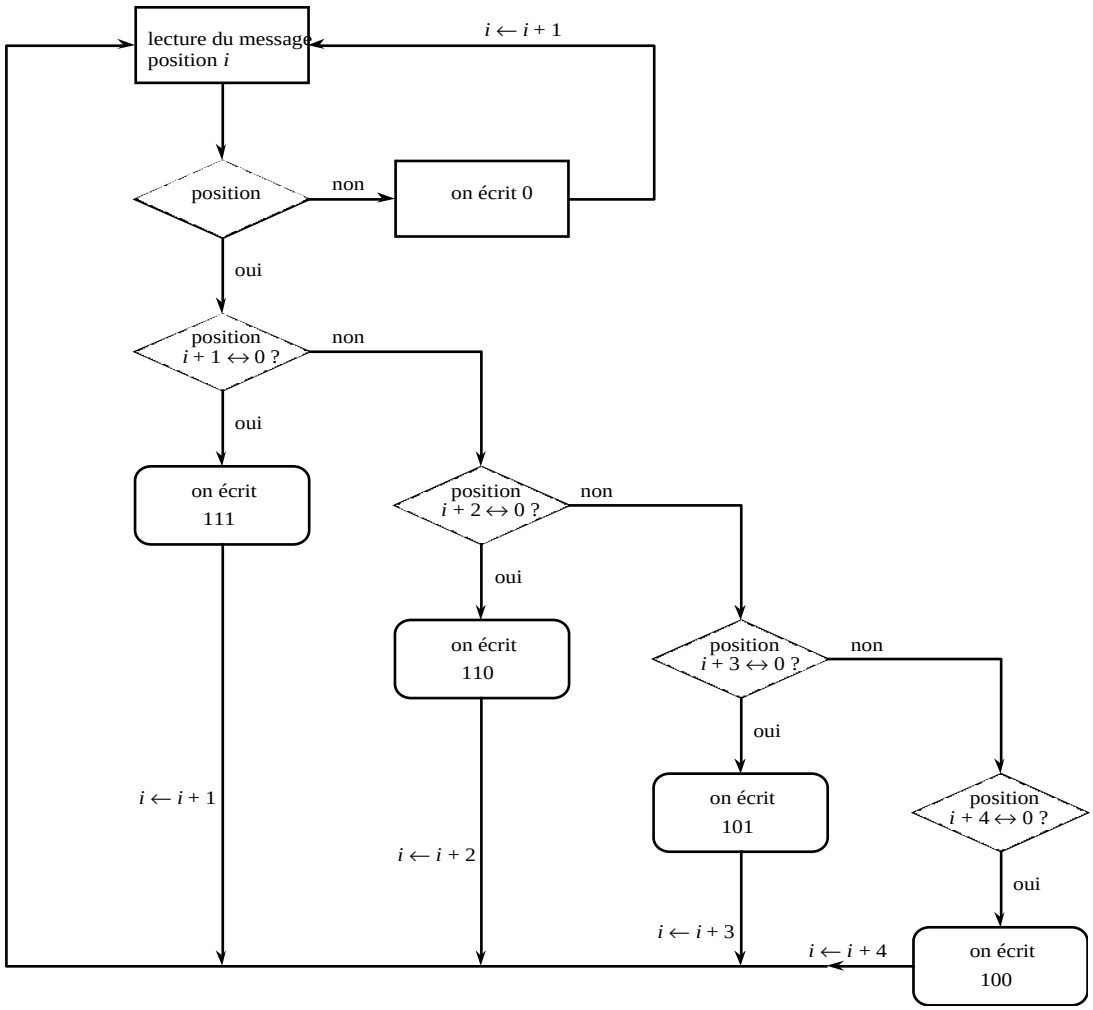
0 100 111 0 110 0 0

donnera

0 1111 10 0 11 0 0

Pour l'ordinateur qui code le message à transmettre, il y a deux possiblités : soit on rencontre un 0 qu'on ne change pas, soit on rencontre un 1 : on regarde les deux positions suivantes pour obtenir un bloc de 3 qu'on remplace par son codage correspondant.

Et voici le schéma de codage du message binaire utilisé par l'ordinateur :



Le bit inverseur.

Nous allons donner une solution qui permet d'éviter à la fois « 1111 » et « 0000 ».

Codage.

Dès la reconnaissance de « 111 » ou de « 000 », l'ordinateur regarde le bit suivant. Si celui-ci est identique aux trois autres, on l'inverse (c'est à dire que le « 0 » devient « 1 » ; « 1 » devient « 0 ») ; sinon on le laisse tel quel.

On rajoute ensuite un bit appelé « bit inverseur ». Ce bit vaut « 1 », si la position précédente est inversée et vaut « 0 » sinon.

[Voici un exemple :]

message initial

0 0 0 0 0 1 1 1 0 1 1 1 1

message codé

0 0 0 1 1 0 1 1 1 0 0 1 1 1 0 1

↑ indique l'inversion du bit précédent

↑ indique la non-inversion du bit précédent

↑ indique l'inversion du bit précédent

Décodage.

[Voici un exemple :]

	identité		identité		identité
codé	0 0 0 1	1	0 1 1 1 0	0	1 1 1 0 1
décodé	0 0 0 0		0 1 1 1 0		1 1 1 1
		↑ inversé		↑ non inversé	↑ inversé

Principe du décodage.

On identifie un des deux blocs « 1110 » ou « 0001 ». Lorsqu'on a trouvé l'un des deux, on regarde le bit suivant (c'est le bit inverseur). Si sa valeur est « 1 », on le supprime et on change le bit précédent ; si sa valeur est « 0 », on le supprime sans rien changer.

[NDLR. Les décodages proposés sont-ils toujours possibles ? Peut-on être sûr que ces blocs proviennent bien du codage, et non d'un mélange avec d'autres bits ?]