Examen 2002/2003 Compression

Le 20/12/2002, Tout document autorisé

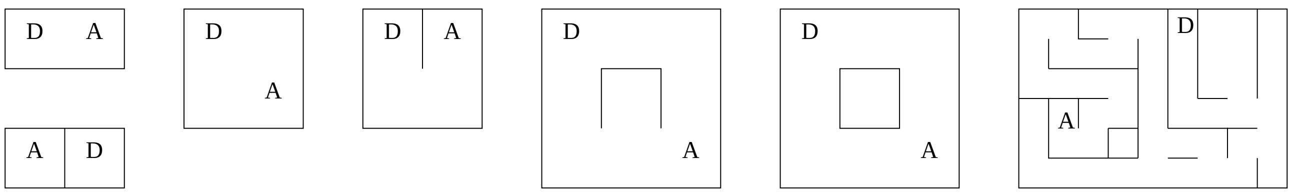
Téléphones cellulaires

Les téléphones cellulaires (portables) communiquent en numérique avec les émetteurs/récepteurs les plus proches, cette liste d'émetteur peut changer lorsque le téléphone est déplacé. Un téléphone portable est un ordinateur, avec un écran, un clavier, une carte "radio" et un système d'exploitation temps réel. Il possède un processeur avec une horloge d'environ 100MHz et n'a pas plus de 256Ko de mémoire. Ces limitations permettent de baisser les prix et la consommation électrique. Évidemment, la bande passante hertzienne utilisée étant très faible il est nécessaire de comprimer les données transmises. **Les questions suivantes s'appliquent dans ce contexte** :

1. Indépendamment de la méthode de compression utilisée pour transmette les données, faites des remarques sur la mise en œuvre de la méthode d'égalisation des probabilités utilisée par le compresseur.
2. Pour de la compression de données, donnez les avantages et inconvénients de LZ77, LZ78, LZW et du *Block sorting*.
3. Le téléphone doit charger une image, l'écran est noir et blanc (ou dispose de très peu de couleurs) et l'image originale est couleur. Qui peut se charger de la conversion ? Listez les avantages et inconvénients des différentes solutions.
4. Les téléphones portables peuvent recevoir des pages WML. Le WML est une variation sur le thème HTML. Le codage HTML est très redondant, proposez un codage plus court, plus rapide à analyser (*parse*) et nécessitant moins de mémoire.
5. Pourquoi n'est ce pas une bonne idée d'avoir une méthode de compression/décompression avec état entre le téléphone et l'émetteur ?
6. Pourquoi la voix sur Internet passe mal, alors que sur les portables il n'y a pas de problème bien que cela soit aussi en numérique ?
7. Pour pouvoir contacter un portable, il faut savoir à tout moment où il est. Il faut donc une base de données donnant le numéro de l'émetteur en fonction du numéro de téléphone. Quel est la taille de cette base de données si l'on veut optimiser la vitesse de mise à jour, ou bien si l'on veut optimiser la place mémoire.
8. La bande passante d'un émetteur est saturée. Un téléphone de plus veut communiquer, que faire ?
9. Comparez les contraintes de compression et de transmission pour une conversation téléphonique, et pour la réception de musique.
10. Comment améliorer la réception de la musique ?

Labyrinthes

Voici quelques labyrinthes complètement quelconques et pas forcément intéressants. D=départ et A=arrivée.



1. Quelles sont les règles générales qui sont exactes pour tous les exemples ci-dessus ?
2. Proposer un codage simple, efficace, rapide d'accès et optimal permettant de représenter tous les labyrinthes respectant les règles que vous vous êtes fixées.
3. Les *bons* labyrinthes doivent respecter certaines règles : ne pas créer de zones inateignables (donc pouvoir être résolu), ne pas faire de rectangles sans cloison mais seulement des couloirs. Pensez-vous qu'il soit possible d'avoir un codage utilisable dans la réalité ne décrivant que ce type de labyrinthe ?
4. Existe-t-il un codage théorique optimal des *bon* labyrinthes ? Si oui lequel ?
5. Donner les avantages et inconvénient du codage du graphe des couloirs au lieu des cloisons.
6. On se limite aux labyrinthes dont la taille en cases est paire dans les deux directions. Quel est l'intérêt de faire un codage du labyrinthe par pavés de 2x2 cases ? Même question si l'on comprime ce codage.
7. Les *bons* labyrinthes vont-ils être comprimés efficacement ?
8. Les *très bons* labyrinthes, ceux qui sont très difficile à résoudre pour un être humain, vont-ils être comprimés efficacement ?
9. Proposez un algorithme simple (3 lignes en bon français) permettant de générer aléatoirement un *bon* labyrinthe.
10. Dans un jeu en réseau un très gros labyrinthe aléatoire doit être transmit à tous les joueurs à chaque nouvelle partie. Comment faire pour transmettre ce labyrinthe en quelques octets ? Ou même 0 octets si les conditions de permettent.

Correction

Comme d'habitude, cela ne sert à rien de recopier son cours dans un examen avec documents autorisés. Vous devez répondre aux questions.

Téléphone.

1. Pour minimiser les ressources utilisées, il faudrai employer une table de Huffman statique. *Remarque : Que cela soit Huffman ou Shanon-Fano, la taille de l'arbre des événements est la même. Remarque : On égalise les probabilités directement sur le son (on gagnerai peu) mais sur la DCT filtrée du son. Remarque : Les algorithmes ne s'exécutant pas en un temps fixe sont difficiles à mettre en œuvre dans un environnement temps réel. Remarque : C'était une question de cours, j'ai explicitement parlé de ce problème en présentant la compression des FAX.*
2. LZ77 est rapide pour décomprimer et utilise une place facilement paramétrable pour le buffer. Toutes les autres méthodes prennent beaucoup plus de temps de décompression et ne sont efficaces qu'avec beaucoup de mémoire.
3. Par le destinataire : perte de bande passante.
   Par l'émetteur : charge l'émetteur, la traduction est faites pour chaque destinataire.
   Par la source : solution optimum, mais elle doit pouvoir savoir si le destinataire est n&b. *Remarque : La question était qui et non comment.*
4. On code le WML (HTML du WAP) comme un arbre ou les tags et attributs sont des noeuds et le texte des feuilles. Le type du tag est codé comme un entier et non en chaîne de caractère. De plus les entiers utilisés peuvent être contextuels (Dans un tag HEAD il n'y a que 7 tags possibles, il n'y a que 2 attributs possibles pour TITLE). Cela prend considérablement moins de place et le décodage est "instantané" car il n'y a pas d'analyse de syntaxe à faire.
5. En cas de perte de donnée, il faut avoir un protocole de correction d'erreur et cela est inaceptable si l'on transfert de la voix en temps réel. De plus il faut gérer un transfert d'état courant quand on change de cellule. *Remarque : Un système à état courant économise de la bande passante car il n'est pas nécessaire de le transmettre. Remarque : La taille de l'état courant est quelconque donc on s'arrangerai pour que cela rentre dans le téléphone sans problèmes. Remarque : Dans la réalité, la transmission du son est faites sans état car on transmet la DCT paquet par paquet (indépendant les uns des autres) et que l'équilibrage de probabilités utilisé est statique donc sans état.*
6. Par ce le QoS n'est pas implémenté sur Internet, on ne peut pas réserver sa bande passante. *Remarque : la transmission entre 2 portables est numérique de bout en bout, car déjà en 1993 la très grande majorité des commutateurs dialoguaient en numérique entres-eux. Remarque : le fait d'avoir une liaison virtuel entre deux points ne réserve pas obligatoirement la bande passante. Remarque : les données circulant sur Internet ne sont pas comprimé par "le réseau", sauf pour les MoDem qui le font de manière* transparente. *Remarque : Le débit sur Internet via modem est supérieur à celui des téléphones portables, le modem sur téléphone fixe 56Kb, le téléphone portable 9.6Kb.*
7. Vitesse de mise à jour : un tableau de numéro d'emetteur indexé par les numéros de téléphones
   Place mémoire : Table de hashage, mais cela n'a d'intérêt que s'il y a plein de numéros de téléphones inutilisés (la table de la première solution est trouée) *Remarque : La version simple stocke 100.000.000 de numéros d'émetteur, même en supposant 4 milliards d'émetteurs en France, cela représente 400Mo ce que je n'estime pas énorme du tout. Remarque : On doit noter tous les téléphones dans cette base, y compris ceux en veilles.*
8. Refuser la nouvelle communication, baisser la qualité de communication de tout le monde, se connecter à une cellule voisine. Toutes ces réponses sont correcte, mais seul la deuxième a un rapport avec la compression.
9. La musique n'est pas transférée en temps réel. Donc on peux bufferiser pour éviter les coupures en cas de perte de la liaison. De plus on peut utiliser un protocol de correction d'erreur car l'utilisateur n'entendra pas la coupure de transmission grace au buffer. On peut utiliser des gros paquets pour la DCT. *Remarque : Rapide et Temps réel sont deux concepts différents, on peut être l'un sans l'autre et inversement ; être temps réel cela indique que l'on réponds en un temps borné prédéfini, être rapide cela indique que la vitesse* **moyenne** *de réponse est grande. Remarque : Plus la fréquence d'échantillonnage est grande plus facilement le signal se compressera car il sera plus lisse/régulier. Remarque : La musique est un signal que l'on veut moins dégrader que la voix, c'est en plus un signal plus riche en fréquences*
10. Avoir un gros buffer, utiliser des gros paquets, faire de la correction d'erreur. On est limité par la puissance du téléphone.

Labyrinthe

1. Ce sont des grilles rectangulaires régulières auquels il manque des segments à l'intérieur et qui contiennent des labels D et A dans deux cases différentes de la grille. *Remarque : plus vous donnez de règles ici, plus les questions suivantes sont complexes et donc dur à répondre. Remarque : la règle du jeu n'était pas demandée, mais c'est vrai que la question était ambigüe.*
2. Les coordonnées de D et A, la taille en cases L, H, les tableaux de booléens *vertical[H][L-1]* et *horizontal[H-1][L]* qui contiennent vrai s'il y a une cloison. *Remarque : ne pas parler d'arbre ici, ce n'est pas un codage rapide d'accès car on doit le parcourir, et il est moins simple qu'un tableau. Remarque : Pour les grand labyrinthes, on tend vers 2 bits par case. Remarque : Coder pour chaque case si elle contient A ou D fait perdre beaucoup de place et ne permet pas de trouver A ou D instantanément.*
3. On code un arbre des couloirs + des cloisons à percer. Si l'on code le numéro de la cloison à percer comme un index dans le tableau des cloisons perçables sans générer de pièce. Si l'arbre de couloir recouvre le rectangle on est alors assuré d'avoir un bon labyrinthe.

Pour ce faire, on part du haut gauche et le chemin est indiqué avec les directives G D H B ( ) cette solution prend plus de place que la solution triviale. On peut l'améliorer en éliminant de cette liste de directive celles qui nous ramène sur une case déjà explorée ou bien qui sont impossibles. De cette manière aucune information inutile n'est codée et l'on est **assuré** d'avoir un bon labyrinthe (sauf si le code est trop court). De plus ce code contient des suites de 0 quand on définit un couloir encadré par deux couloirs déjà défini. Dans certain cas, la liste des directives possibles peut être réduite à l'ensemble vide, dans ce cas le codage du couloir ne prend aucune place ! A ce codage de l'arbre des couloirs il faut ajouter une liste de cloisons à faire sauter si l'on désire faire des couloirs en boucle.

Remarque : Un bon labyrinthe se doit de contenir des couloirs en boucles afin que l'algorithme consistant à suivre une cloison ne vous amène pas forcément à la sortie. Remarque : Ce codage n'est pas optimal car il y a plusieurs façon différentes de coder le même labyrinthe.

4. Vous numérotez mathématiquement tous les bons labyrinthes, vous n'avez donc ensuite plus qu'à coder le numéro du labyrinthe qui vous intéresse.
5. Par exemple, pour un jeu en 3D. Quand on connait la position du joueur, on peut facilement afficher les cloisons près de lui en premier si l'on a le graphe des couloirs. Si on a le graphe des cloisons cela nécessite de parcourir tout le graphe. *Remarque : si le graphe des cloisons (sans le tour) est un ensemble d'arbres touchant le bord moins de 2 fois alors il y a une solution. Si le graphe des couloirs n'est pas connexe alors il n'y a pas de solution, si c'est un arbre alors il n'y a pas de chemin circulaire (dommage).*
6. En codant des pavés 2x2, on peut facilement faire disparaître le cas de la salle 2x2 vide et celui d'une case complètement fermée. On code la case par 8 bits (8 cloisons possibles) un compresseur de texte éliminera automatiquement les cas qui n'apparaissent jamais dans le labyrinthe car il travaille sur des caractères (8 bits).
7. Non, car il n'y aura pas de grande zones pleine de cloisons ou pleines de vide. Les cas qui se comprime le mieux ne sont pas des bons labyrinthes.
8. Si un bon labyrinthe contient des répétitions visuelles il sera plus facilement solvables. Par exemple pleins de couloirs parallèles ou des zones identiques. Donc le compresseur ne devrait pas trouver de répétitions dans un très bon labyrinthe, ils se comprimeront donc très mal. *Remarque : Ceci n'est pas vrai pour les labyrinthes grandeur réel ou la personne se déplace à l'intérieur car les similarités l'empêcheront de prendre des repères.*
9.
   - On met des cloisons partout une grille.
   - On enlève aléatoirement une cloison interne (sans créer de salle), tant qu'il y a une zone close.
   - On choisi D et A les plus éloignés l'un de l'autre.
10. On transmet la graine de la série aléatoire, ou bien on peut utiliser l'heure précise du début de la partie si elle est connue de tout le monde.