Correction

EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE – LAUSANNE
POLITECNICO FEDERALE – LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY – LAUSANNE

Faculté Informatique et Communications
Cours ICC aux sections MA et PH
Chappelier J.-C.

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

INFORMATIQUE, CALCUL & COMMUNICATIONS

Sections MA & PH

Correction de l'examen intermédiaire III

22 décembre 2017

SUJET 1

Instructions :

- Vous disposez d'une heure quinze minutes pour faire cet examen (15h15 - 16h30).
- L'examen est composé de 2 parties : un questionnaire à choix multiples, à 12 points, prévu sur 40 minutes, et une partie à questions ouvertes, à 8 points, prévue sur 35 minutes.
  Mais vous êtes libre de gérer votre temps comme bon vous semble.
- **AUCUN DOCUMENT N'EST AUTORISÉ, NI AUCUN MATÉRIEL ÉLECTRONIQUE**.
- Pour la première partie (questions à choix multiples), chaque question n'a qu'une seule réponse correcte parmi les quatre propositions.
  Indiquez vos réponses en bas de **cette** page en écrivant *clairement* pour chaque question une lettre majuscule parmi A, B, C et D. (Vous êtes autorisé à dégrafer cette page)
  Aucune autre réponse ne sera considérée, et en cas de rature, ou de toute ambiguïté de réponse, nous compterons la réponse comme fausse.
- Pour la seconde partie, répondez directement sur la donnée, à la place libre prévue à cet effet. Aucune feuille supplémentaire ne sera considérée.
- Toutes les questions comptent pour la note finale.

Réponses aux quiz :

Reportez ici *en majuscule* la lettre de la réponse choisie pour chaque question, sans aucune rature.

1	2	3	4	5	6	7	8	9	10	11	12
B	D	A	C	B	D	B	C	C	D	B	A

PARTIE QUIZ

Question 1) En utilisant la politique de remplacement de cache LRU vue en cours, et sachant que la mémoire cache peut contenir 3 blocs de 2 mots, combien de défauts de cache génère la séquence d'accès mémoire suivante (adresses de mots) en partant d'une cache vide :

```
11  12  10  11   5  13   7  11  10   6  12
```

Les adresses commencent à 0 et les blocs sont toujours alignés sur les multiples de 2 en mémoire centrale.

A] 4 **✔B]** 5 **C]** 6 **D]** 8

La séquence des adresses de *blocs*, avec les défauts de cache :

```
10  12  10  10   4  12   6  10  10   6  12
 *   *           *       *   *
```

Question 2) Sur un ordinateur avec un processeur à 64 bits, combien de temps faut-il pour que le *processeur* lise séquentiellement tous les octets d'un fichier de 16 Mo sachant que :

- le disque transfert 5 mots par μs à la mémoire centrale ;
- la cache est constituée de 4 blocs de 8 mots ;
- le transfert d'un bloc en mémoire cache prend 240 ns ;
- la communication cache–processeur est de 15 ns par mot.

A] 400 ms **B]** 400 μs **C]** 490 μs **✔D]** 490 ms

En supposant toutes les opérations séquentielles (pas de parallélisme), en ms : temps lecture du ficher depuis le disque : $16/(8\times 5)\times 10^3 = 400$, défauts de cache ($16/(8\times 8)$ blocs) : $240\times 16/(8\times 8)$, lecture des mots par le processeurs : $15\times 16/8$.

Question 3) Pour un programme qui utilise tellement de variables qu'elles sont éparpillées dans la mémoire centrale et utilise systématiquement deux de ces variables pour calculer leur produit, il vaut mieux avoir :

✔A] de petits blocs car la localité temporelle est importante.
B] de grands blocs car la localité temporelle est importante.
C] de grands blocs car la localité spatiale est importante.
D] de petits blocs car la localité spatiale est importante.

Question 4) Quelle est, en fonction de $n = \min(\textbf{taille}(L_1), \textbf{taille}(L_2))$, la complexité de l'algorithme suivant :

késeksa

entrée : *Deux listes L_1 et L_2*
sortie : *? ? ?*

```
n ⟵ min(taille(L1),taille(L2))
L3 ⟵ (0, 0, 0)
Pour i de 1 à n
   Pour j de 1 à n
      k ⟵ 0
      Tant que k + i ≤ n et k + j ≤ n et L1[k + i] = L2[k + i]
         k ⟵ k + 1
         Tant que k > L3[1]
            L3 ⟵ (k, i, j)
Sortir : L3
```

A] $O(n)$ **B]** $O(n^2)$, mais pas $O(n)$. **✔C]** $O(n^3)$, mais pas $O(n^2)$. **D]** $O(n^4)$, mais pas $O(n^3)$.

suite au dos ☞

Question 5) Considérez le code assembleur suivant :

```
 1: charge   r2,  0
 2: charge   r3,  0
 3: charge   r4, r3
 4: somme    r3, r3, 1
 5: somme    r4, r4, 1
 6: cont_ppe r1, r4, 9
 7: somme    r2, r2, r4
 8: continue 5
 9: somme    r2, r2, r3
10: cont_pp  r3, r0, 3
```

où l'instruction « `cont_ppe a, b, N` » effectue le test « `a`$\leq$`b` » et l'instruction « `cont_pp a, b, N` » effectue le test « `a`$<$`b` ».

Lequel de ces algorithmes correspond au code ci-dessus (avec n chargé dans `r0` et m dans `r1`) :

A.

algoA

entrée : *deux entiers* $1 \leq n \leq m$
sortie : S

$S \longleftarrow 0$
Pour i de 1 à $n-1$
  Pour j de 1 à m
    $S \longleftarrow S + j$
  $S \longleftarrow S + i$
Sortir : S

✔B.

algoB

entrée : *deux entiers* $1 \leq n \leq m$
sortie : S

Si $n = 1$
  Sortir : $1 + \frac{(m-1)\cdot m}{2}$
$s \longleftarrow n$
Pour i de $m-1$ à n en descendant (de 1 en 1)
  $s \longleftarrow s + i$
Sortir : $s +$ **algoB**$(n-1, m)$

C.

algoC

entrée : *deux entiers* $1 \leq n \leq m$
sortie : S

$S \longleftarrow 0$
$i \longleftarrow 0$
Tant que $i < n$
  $j \longleftarrow i$
  Tant que $j < m$
    Si $j < m$
      $S \longleftarrow S + j$
    Sinon
      $S \longleftarrow S + i$
Sortir : S

D.

algoD

entrée : *deux entiers* $1 \leq n \leq m$
sortie : S

$S \longleftarrow 0$
Pour i de n à 1 en descendant (de 1 en 1)
  Pour j de $m-1$ à i en descendant (de 1 en 1)
    $S \longleftarrow S + i$
  $S \longleftarrow S + j$
Sortir : S

Les bornes de l'algorithme A ne sont pas correctes : la boucle en i devrait aller jusque n et celle en j de i à $m-1$ (à voir en testant le cas simple $n = 1$).

L'algorithme B est une écriture récursive de ce que calcule le programme :

$$S(n,m) = \sum_{i=1}^{n} \left(\left(\sum_{j=i}^{m-1} j \right) + i \right) = \sum_{i=1}^{n-1} \left(\left(\sum_{j=i}^{m-1} j \right) + i \right) + \sum_{j=n}^{m-1} j + n = n + \sum_{j=n}^{m-1} j + S(n-1,m)$$

Dans l'algorithme C, il manque les incréments de i et j dans leur boucle respective (boucles infinies).

Dans l'algorithme D, les deux lignes « $S \longleftarrow S + i$ » et « $S \longleftarrow S + j$ » ont été inversées (aucun sens !).

Question 6) Soit X le signal défini par :

$$X(t) = \sin(6\pi t + \frac{\pi}{4}) + 3\sin(30\pi t + \frac{\pi}{3}) + 2\sin(12\pi t).$$

X est filtré par un filtre passe-bas idéal de fréquence de coupure $f_c = 12$ Hz, puis échantillonné à une fréquence $f_e = 14$ Hz. A partir de ces échantillons, on reconstruit le signal Y en utilisant la formule de reconstruction vue en cours.

Quelle est la forme mathématique du signal $Y(t)$?

A] aucune des trois autres car il y a repliement de spectre (effet stroboscopique).

B] $X(t)$.

C] $\sin(6\pi t + \frac{\pi}{4})$.

✔**D]** $\sin(6\pi t + \frac{\pi}{4}) + 2\sin(12\pi t)$.

Question 7) Si je veux transmettre vos notes au service académique (SAC) à la fin de l'année, mais souhaite qu'elles restent lisibles (par exemple par vous), alors j'assure :

A] l'intégrité des notes en les chiffrant avec la clé publique du SAC.

✔**B]** l'intégrité et l'identité de l'auteur des notes en les signant avec ma clé privée.

C] l'identité de l'auteur des notes en les chiffrant avec la clé publique du SAC.

D] la confidentialité des notes en les chiffrant avec ma clé privée.

Question 8) Dans un système de cryptographie à clé publique,

A] on ne peut pas avoir à la fois de la confidentialité et de la responsabilité.

B] chaque participant possède au moins deux clés et utilise sa propre clé privée pour envoyer des messages confidentiels.

✔**C]** chaque utilisateur a une clé publique et une clé privée qu'il est, à l'heure actuelle, pratiquement impossible de retrouver avec la clé publique.

D] aucune confidentialité n'est possible puisque la clé est publique.

Question 9) Dans un système RSA, j'utilise la clé publique $(7, 33)$ et ma clé privée est $(3, 20)$. Je veux envoyer de façon confidentielle le message `00001000` à un destinataire de clé publique $(5, 69)$. J'envoie alors :

A] `00001100` **B]** `00000010` ✔**C]** `00111110` **D]** `00010001`

Remarque : $2^5 = 32 = -1 \mod 33$, $2^6 = 64 = 4 \mod 20$, $2^6 = 64 = -5 \mod 69$ et $69 \times 3 = 207$.

$$8^5 = 2^{15} = 2^6 \times 2^6 \times 2^3 = 25 \times 8 = 200 = (-7) = 62 \mod 69$$

Question 10) Dans le même système RSA, je veux envoyer de façon ***non*** confidentielle mais intègre et signée le même message que précédemment au même destinataire. J'envoie alors :

A] `00001100` **B]** `00000010` **C]** `00111110` ✔**D]** `00010001`

$$8^3 = 2^9 = 2^5 \times 2^4 = (-1) \times 16 = 17 = 16 + 1 \mod 33$$

Question 11) Dans un réseau TCP/IP, si un paquet est perdu lors de l'échange d'emails :

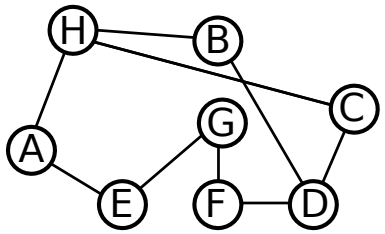
A] ce n'est pas grave car les emails sont redondants.

✔**B]** le paquet est renvoyé par l'ordinateur émetteur.

C] le paquet est renvoyé par le dernier routeur.

D] c'est qu'il a été perdu par la couche TCP car la couche IP garantit aucune perte de paquet.

Question 12) Laquelle des lignes suivantes fait partie de la table de routage de G pour ce réseau IP :



✔**A]**

dest.	dir.	dist.
H	E	3

B]

dest.	dir.	dist.
B	E	3

C]

dest.	dir.	dist.
F	B	3

D]

dest.	dir.	dist.
B	E	4

(vous pouvez utiliser cette partie pour répondre aux exercices suivants, mais veuillez s.v.p. préciser le numéro de la question traitée)

PARTIE EXERCICES

1 – Chant (de Noël ?) [3 points]

Un enfant de vos connaissances veut enregistrer, et envoyer par message, un chant à sa grand-mère. Fort(e) de vos connaissances en ICC, vous l'aidez à construire une transmission optimale.

Question 13) **[1 point]** Sachant que sa voix se situe entre 1000 et 3500 Hz (au dessus il n'y a que du bruit) et que pour des raisons techniques, il n'est pas possible d'acquérir plus que 8000 échantillons par seconde, quelle procédure lui conseillez-vous :

- filtrage (passe-bas) : oui ou non ? ☞**oui**
- Si oui :
  - avant ou après échantillonnage ? ☞**avant**
  - avec quelle fréquence de coupure ? ☞ **un peu au dessus** (mais assez proche) **de 3500 Hz** (par exemple $3500 + 1$ Hz)
- À quelle fréquence conseillez-vous d'échantillonner ? Pourquoi ? ☞**entre strictement plus que deux fois la fréquence de coupure et 8000 Hz** (mais le mieux étant, en pratique, de prendre un peu de marge ; p.ex. 7100 Hz). Cela permet de respecter les contraintes et de reconstruire parfaitement le signal (strict. sup. à 2 fois la bande passante)
- Le chant pourra-t-il être correctement reconstruit à l'arrivée ? Pourquoi ? ☞**oui**, cf ci-dessus. Par contre sans filtrage c'est totalement impossible puisque le bruit créera des perturbations par repliement de spectre.

Question 14) **[1 point]** Vous conseillez ensuite de compresser, sans perte, le signal échantillonné (et le message qui le contient). Le message d'origine complet considéré bit à bit a une entropie de 0.9 bit et considéré octet par octet, une entropie de 5.51 bits.

Sachant que ce message (d'origine, complet) a une taille de 10 Ko, quelle taille pouvez-vous espérer après compression (sans autre donnée que celles fournies ici) ?

Justifiez votre réponse.

Question 15) **[1 point]** Comme le message est posté sur un réseau public, vous conseillez de l'encrypter en utilisant RSA. Pour simplifier, supposons que la grand-mère a pour clé publique $(1021, 6557)$ et que l'enfant a $(2945, 8448)$ pour clé privée et $(7553, 8633)$ pour clé publique.

On transmet le message octet par octet. Après compression, le message commence par « `10010011` ».

Quel est alors le premier octet envoyé ? Exprimez le résultat sous la forme « $a^b \mod c$ » avec a, b et c des nombres entiers *écrits en décimal*.

---

14) bit à bit, le message ne nous apporte rien (ni le th. de Shannon) ;
octet par octet par contre, l'entropie (5.51 bits) étant significativement plus petite que 8 bits, on peut espérer (bornes des deux parties du th. de Shannon) entre 5.51 et $5.51 + 1$ bits par octet, soit au final entre $5.51 \cdot 10/8$ et $(5.51 + 1) \cdot 10/8$ Ko, soit environ 7.5 Ko.

15) Ce que l'on cherche ici c'est de la *confidentialité*, uniquement, pas nécessairement de l'intégrité (risque de modification du message quasi nul (qui s'y intéresserait ?) et la grand-mère reconnaitra bien par elle-même si c'est bien le chant de sa petite-fille ou non). Il suffit donc de simplement encrypter le message (avec la clé publique de la grand-mère) :

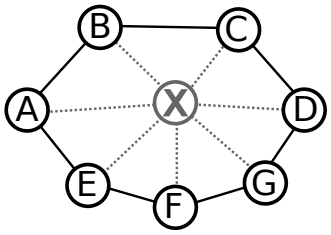
$$\texttt{10010011}^{1021} \mod 6557 = 147^{1021} \mod 6557$$

($a = 147$, $b = 1021$ et $c = 6557$).

(Note : en RSA, on travaille modulo n, donc forcément avec des nombres entiers positifs.)

2 – Route (de Noël ?) [5 points]

On considère le réseau TCP/IP suivant :



Question 16) **[1 point]** Donnez la ligne de la table de routage de A correspondant au nœud D lorsque le nœud X est présent, puis lorsqu'il est absent.

Question 17) **[2 points]** On envoie un message de A vers D (X absent). Ce message est décomposé en 50 paquets. Sachant que

- chaque nœud met 10 ms à transmettre un paquet au nœud suivant ;
- on négligera tout autre temps ;
- en moyenne 1 paquet sur 4 est malheureusement perdu ;

quel est (en moyenne) le temps total de transmission de ce message, lorsque X est absent ?

Justifiez votre réponse.

Question 18) **[2 points]** La machine réceptrice (D) a un processeur de 64 bits et une mémoire cache (LRU) de 5 blocs de 2 mots. Sachant qu'un paquet réseau correspond à 4 mots, combien de défauts de cache fait cette machine pour correctement lire (dans le bon ordre) un message dont les paquets sont arrivés dans l'ordre 2, 1, 5, 3, 4 ?

On admettra que la machine doit lire l'entièreté du paquet pour savoir de quel paquet il s'agit et que tous les paquets résident en mémoire centrale dès réception, à des adresses successives ; mais on ne connaît bien sûr pas leur ordre avant de les avoir lus une première fois.

Justifiez votre réponse.

---

16)

avec X :

destination	route	distance
D	X	2

sans X :

destination	route	distance
D	B	3

17) À chaque fois que A envoie x paquets à D :

- 1/4 de ces paquets est perdu, ce dont A se rend compte par le fait que D n'a pas envoyé de paquet d'acquittement ;
- **mais aussi** 1/4 des $(1 - 1/4) \cdot x$ paquets d'acquittement renvoyés par D dont perdus, donc A considère les paquets de départ correspondants aussi comme perdus.

Donc au final pour x paquets, A doit renvoyer $x/4 + x(1-1/4)/4 = (2 \times 4 - 1)x/4^2 = 7x/16$ paquets.

Et cela tant que A doit renvoyer des paquets, donc un nombre $n + 1$ de fois tel que $(7/16)^n \times 50 = 1$, soit $\log 50 / \log(7/16) + 1 = 5 + 1$ fois (pas demandé).

Du point de vue du temps, comme il y a 3 nœuds de transmission, le premier paquet met 3×10 à arriver à D et son acquittement 3×10 à revenir à A (latence). Ensuite (débit), un paquet (mais aussi son acquittement) arrive toutes les 10 ms. Donc globalement avant de renvoyer quoique ce soit, A

attend $6 \times 10 + (x-1) \times 10 = (x+5) \times 10$ ms.[1]

Et donc, globalement, la transmission de 50 paquets prend :

$$10\times\Big(50+5+50\times 7/16+5+50\times(7/16)^2+5+...+1+5\Big) = 10\times\left(50\times\sum_{i=0}^{5}(7/16)^i+5\times(5+1)\right)$$

(ce qui, avec une calculette, mais ce n'était évidemment pas attendu ici, donne au final environ 1185 ms).

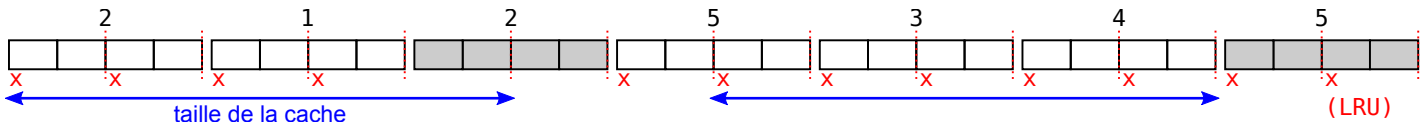
Une réponse moins précise mais acceptée ici car elle garde les aspects essentiels des communications TCP/IP (aller-retour, donc fateur 2 pour simplifier, et pertes aussi sur la seconde, troisième, ..., communication) pourrait être (considérant les communications de chaque paquet comme séquentielles nœud après nœud, c.-à-d. en ignorant le parallélisme des nœuds) :

$$3 \times 10 \times 50 \times 2 \times \Big(1 + 1/4 + (1/4)^2 + ...\Big)$$

qui donnerait :

$$\simeq 6 \times 10 \times (50 + 12.5 + 3 + 1) \simeq 4000 \text{ ms}$$

18) Pour remettre toute la transmission dans l'ordre, la machine lit alors les paquets dans l'ordre suivant : 2, 1, 2, 5, 3, 4, 5 (stratégie optimale ; on peut aussi considérer tout relire deux fois, mais c'est moins bien).



Ce faisant, elle fait au final **12 défauts de cache** : 2 par lecture de paquet, aucun pour relire 2 (qui est encore en cache), mais bien 2 pour relire le paquet 5 (qui n'est, juste pas, plus en cache).

1. Deux remarques : (1) en réalité A attendrait certainement plus longtemps, mais cela n'est pas du tout considéré ici puisque nous avons clairement dit que tout autre temps était négligé ; (2) on pourrait ne pas considérer le temps d'acquittement des tous derniers paquets de la communication globale, mais c'est un détail ici.