

ANNALE D'EXAMENS 1^o ANNEE 08 - 09

ALGORITHMIQUE	-----	PAGE	2
ANALYSE	-----	PAGE	16
ARCHITECTURE	-----	PAGE	20
ECONOMIE	-----	PAGE	34
INTRODUCTION AUX RESEAUX	-----	PAGE	40
LOGIQUE	-----	PAGE	43
LOGICIEL DE BASE	-----	PAGE	48
METHODES NUMERIQUES	-----	PAGE	51
ONDES ET TRANSMISSIONS	-----	PAGE	55
PROBABILITEES	-----	PAGE	59
RECHERCHE OPERATIONNELLE	-----	PAGE	67
THEORIE DE LANGAGES	-----	PAGE	73
TRAITEMENT DU SIGNAL	-----	PAGE	83

Algorithmique et structures de données : examen de première session

ENSIMAG 1A

Année scolaire 2008–2009

Consignes générales :

- Durée : 3 heures. Tous documents et calculatrices autorisés.
- Le barème est donné à titre indicatif. Il est sur 15 points, la note de chacun des 2 TP étant ramenée sur 2.5 points.
- Les exercices sont indépendants et peuvent être traités dans le désordre.
- La syntaxe Ada ne sera pas un critère déterminant de l'évaluation des copies. En d'autres termes, les correcteurs n'enlèveront pas de point pour une syntaxe Ada inexacte, mais compréhensible (pensez à bien commenter votre code!).
- **Merci d'indiquer votre numéro de groupe de TD et de rendre votre copie dans le tas correspondant à votre groupe.**

Exercice 1 : B-arbres (10 pts)

Un *B-arbre* est une structure de données permettant d'implémenter un dictionnaire de manière efficace. Il s'agit d'une structure hybride mêlant les structures d'arbres et de listes (ou de tableaux). En effet, chaque noeud de l'arbre contient plusieurs clés du dictionnaire. Un B-arbre peut être vu comme une généralisation d'un arbre binaire de recherche au sens où les valeurs des clés stockées dans les noeuds le sont de manière imbriquée.

Un *B-arbre de degré d*, $d \geq 2$, est un arbre (pas forcément binaire) tel que :

1. chaque noeud x comporte quatre informations :
 - un entier $k \geq 1$: le nombre de clés stockées dans ce noeud ;
 - une liste de k clés, généralement stockées par ordre croissant : $cle_1(x) \leq cle_2(x) \leq \dots \leq cle_k(x)$;
 - un booléen *EstFeuille*, indiquant si x est une feuille de l'arbre ou non ;
 - un pointeur pour chaque noeud fils de x , dans le cas où x n'est pas une feuille ;
2. le nombre de fils d'un noeud x est exactement $k + 1$ (sauf bien sûr si x est une feuille) ;
3. $d \leq k + 1 \leq 2d$, excepté pour la racine de l'arbre qui peut avoir moins de d fils ;
4. les valeurs des clés d'un noeud x et les valeurs des clés de ses fils sont intercalées, voir figure 1 pour un exemple : les clés du premier fils de x ont des valeurs inférieures à la valeur de la première clé de x , les clés du deuxième fils de x ont des valeurs comprises entre les valeurs de la première et de la deuxième clés de x , etc. ;

5. les feuilles sont toutes situées à la même hauteur dans l'arbre, c'est-à-dire au dernier niveau.

La figure 1 montre un exemple de B-arbre.

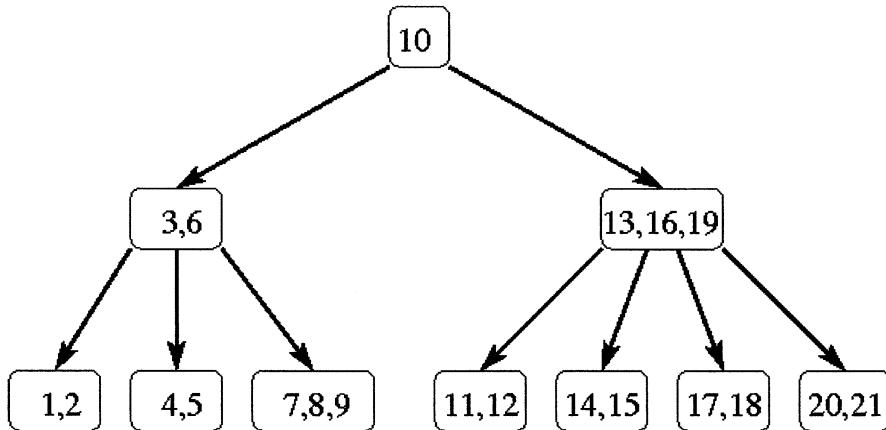


FIG. 1 – Exemple d'un B-arbre.

L'objectif de cet exercice est d'étudier comment implémenter les trois opérations standards que sont la recherche, l'insertion et la suppression d'une clé pour cette structure de données.

1 Questions générales

Question 1 Pour quel(s) degré(s) d l'arbre de la figure 1 est-il un B-arbre valide ? Justifier.

Question 2 Dessiner les quatre B-arbres de degré supérieur ou égal à 2 et dont les clés ont les valeurs $\{ 1, 2, 3, 4, 5 \}$.

Question 3 Définir, en Ada, un type **B-Arbre**. On suppose connus la valeur (fixe) de d et le type **Cle**.

Question 4 Soit n le nombre de clés d'un B-arbre de degré d , et h sa hauteur. Montrer que

$$h \leq \log_d \frac{n+1}{2} \quad (1)$$

On rappelle que la hauteur de l'arbre vide est fixée à -1 .

2 Recherche d'une clé

On s'intéresse dans cette partie à la recherche d'une clé dans un B-arbre.

Question 5 Décrire en français le principe d'un algorithme **récursif** de recherche d'une clé dans un B-arbre.

Question 6 Ecrire la procédure Ada **Recherche(A : in B-Arbre ; C : in Cle ; A' : out B-Arbre ; I : out Integer)** qui implémente cet algorithme.

Cette fonction renvoie à la fois un pointeur vers le nœud contenant la clé recherchée et un entier indiquant la position de la clé dans la liste des clés stockées en ce nœud (exemple : la recherche de la clé 8 dans l'arbre de la figure 1 renvoie un pointeur vers le nœud contenant les clés 7, 8 et 9, ainsi que l'entier 2). Si la clé recherchée n'est pas présente dans l'arbre, la fonction renvoie le pointeur **NULL** ainsi qu'un entier quelconque.

Question 7 Quel est le coût asymptotique en pire cas de cet algorithme, en fonction de n et de d ?

Indication : utilisez la relation (1) de la question 4.

3 Insertion d'une clé

L'insertion d'une clé dans un B-arbre est assez complexe, car il y a de nombreuses propriétés à conserver. La figure 2 montre un exemple de mise à jour d'un B-arbre de degré 3 lors de l'insertion successive de quatre clés.

Comme pour un arbre binaire de recherche, l'insertion se fait en cherchant à partir de la racine la feuille où ajouter la clé, en fonction des valeurs des clés des nœuds traversés. Dans le cas où la feuille où on souhaite insérer la clé est complète, c'est-à-dire qu'elle contient déjà $2d - 1$ clés, il faut "séparer" cette feuille en deux. Cette séparation se fait en remontant la clé médiane vers le nœud père puis en remplaçant la feuille par deux feuilles contenant chacune $d - 1$ clés (respectivement les plus petites et les plus grandes), voir la figure 3 pour un exemple. La nouvelle clé est ensuite insérée dans la feuille adéquate. Mais le problème peut alors se propager : le nœud père peut devenir complet à son tour. Afin de remédier à ce problème, les nœuds complets sont systématiquement séparés (de la même manière) lors de la traversée à partir de la racine.

La procédure globale peut s'écrire de la manière suivante :

```
procedure Insertion(A: in out B-Arbre; C : in Cle) is
    A': B-Arbre;
begin
    if A.K = 2*D-1 then
        -- Si la racine est un noeud complet, on doit la séparer
        --
        -- Dans ce cas, allocation d'un nouveau noeud A', --
        -- avec k = 0, EstFeuille = faux et un unique fils qui est A --
        Separation(A,A',1); -- 1 indique que A est le premier fils de A'
        InsertionNoeudIncomplet(A',C);
    else
```

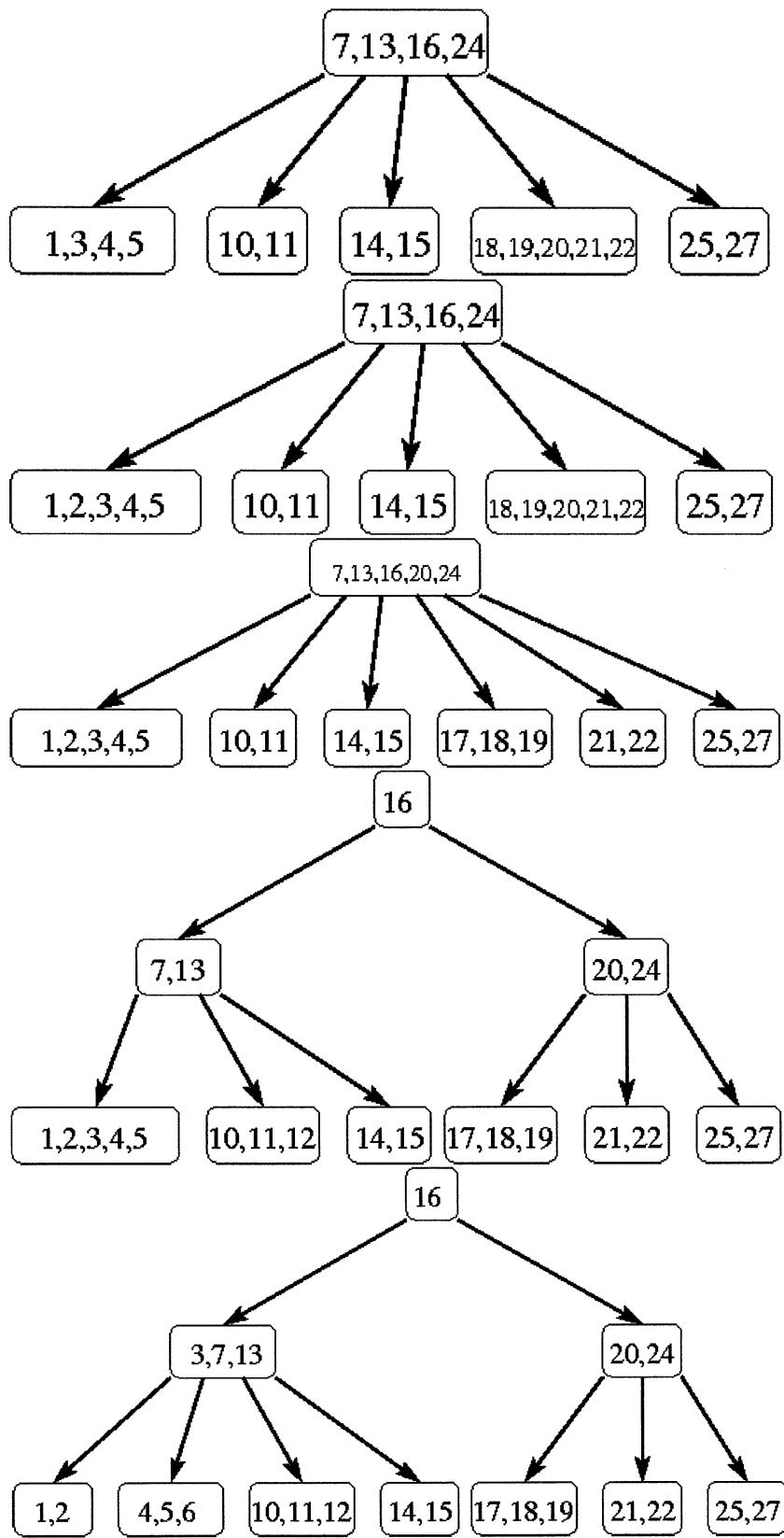


FIG. 2 – Mise à jour d'un B-arbre de degré 3 après insertion successive des clés 2, 17, 12 et 6.

```

InsertionNoeudIncomplet(A,C);
end if;
end Insertion;

```

Cette procédure permet de gérer le cas particulier où la racine de l'arbre est complète, et où il faut donc créer une nouvelle racine. La procédure récursive parcourant l'arbre est la procédure `InsertionNoeudIncomplet(A : in out B-Arbre ; C : in Cle)`.

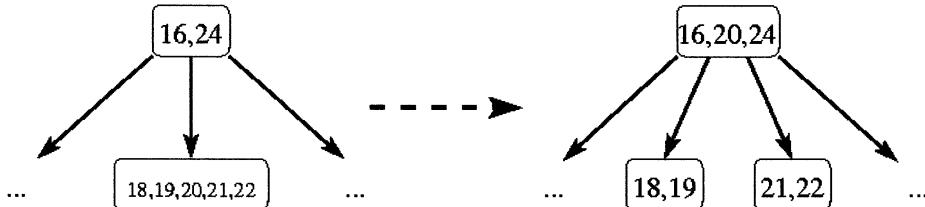


FIG. 3 – Séparation d'un nœud complet.

Question 8 Dessiner les mises à jour d'un B-arbre de degré 2, initialement vide, lors de l'insertion successive des clés 2, 9, 8, 4, 1, 5, 3, 6 et 7.

Question 9 Ecrire en Ada la procédure `Separation(A,A' : in out B-Arbre ; I : in Integer)` qui sépare un nœud complet selon la technique décrite ci-dessus.

Cette procédure prend en paramètre un pointeur `A` vers le noeud, un pointeur `A'` vers son père et un entier `I` indiquant la position du nœud parmi les fils de son père (le noeud étudié est le `I`ème fils de son père). Vous pouvez supposer connue, à condition de bien les définir, toute procédure ou fonction de recopie, d'insertion, de suppression ou de recherche dans une liste ou un tableau.

Question 10 Ecrire en Ada la procédure récursive `InsertionNoeudIncomplet(A : in out B-Arbre ; C : in Cle)` qui insère une clé `C` dans un B-arbre `A` de degré `D`. On suppose que le noeud pointé par `A` n'est pas complet. En revanche, les fils de ce noeud peuvent éventuellement être complets.

Vous pouvez supposer connue, à condition de bien les définir, toute procédure ou fonction de recopie, d'insertion, de suppression ou de recherche dans une liste ou un tableau.

Question 11 Quel est le coût asymptotique en pire cas d'une insertion, en fonction de `n` et de `d`?

Indication : utilisez la relation (1) de la question 4.

4 Suppression d'une clé

L'algorithme de suppression d'une clé est encore plus complexe et ne sera pas traité dans cet examen.

5 Conclusion

Question 12 A quel type d'arbre binaire de recherche vu en cours les B-arbres vous font-ils penser ? Pourquoi ?

Question 13 Quel(s) avantage(s) voyez-vous à utiliser les B-arbres pour gérer un dictionnaire, plutôt que les différents types d'arbres binaires de recherche vus en cours ?

Exercice 2 : Points les plus proches (5 pts)

On se propose de trouver un algorithme rapide pour la recherche des 2 points les plus proches dans un nuage de n points dans le plan. Cet algorithme sera de type “diviser pour régner”.

Les points sont triés une fois pour toutes dans deux tableaux, l'un X par abscisse croissante, l'autre Y par ordonnée croissante. A chaque étape sont passés en paramètres de la fonction, les points P , les deux tableaux X et Y de ces points triés respectivement par abscisse et ordonnée croissantes (ceci se fait à partir de la liste préalablement triée et ne nécessite donc pas un nouveau tri).

On partitionne P en deux parties égales P_1 et P_2 par une ligne verticale Λ . On résout sur P_1 et P_2 , ce qui donne deux points dans P_1 et une distance δ_1 , et deux points dans P_2 et une distance δ_2 . Soit $\delta = \min\{\delta_1, \delta_2\}$. Soit la distance minimum pour les points de P est δ , soit ce sont deux points à cheval sur la frontière Λ qui donnent la plus courte distance.

On prend une bande de largeur 2δ centrée sur la droite Λ .

Question 1 Montrer que pour chaque point p dans cette bande, le rectangle de hauteur δ et de largeur 2δ centré en p ne peut contenir plus de 6 points de P . Pour chaque point de la bande, combien au maximum de distances faudra-t-il calculer ? Borner en fonction de $|P|$ (nombre de points de P) le temps nécessaire à l'exploration de cette bande.

Question 2 Ecrire en Ada la fonction `ExploreBande(Lambda : in Float ; Delta : in Float) return Float` d'exploration de la bande (`Lambda` représente l'abscisse de Λ). Cette fonction renvoie la distance minimum entre deux points de la bande. `X` et `Y` sont des variables globales.

On pourra utiliser les types suivants :

```
type St_Point ;
type Point is access St_Point ;

type St_Point is record
    Abscisse, Ordonnee: Float ;
    IndX: Natural ; -- indice du point dans X
    IndY: Natural ; -- indice du point dans Y
```

```
end record ;  
  
X,Y: array(1..N) of Point ;
```

Question 3 En déduire la complexité de cet algorithme de calcul des points les plus proches.

Question 4 [Bonus] Si on trie dans chaque appel les points, obtient-on la même complexité ? La réponse étant clairement NON, quelle est-elle ? (Pas facile, utiliser le théorème magique de l'appendice du polycopié).

Algorithmique 1

Durée : 3h

Machines électroniques interdites

Tous documents papiers autorisés

Les deux parties du sujet sont indépendantes. Veuillez rendre ces deux parties sur des feuilles séparées.

Veuillez respecter les notations introduites dans l'énoncé. Il est inutile de paraphraser l'énoncé dans vos réponses, mais des explications avec dessins sur votre code sont les bienvenues.

Dans l'ensemble de cet examen, on travaille avec le type **Liste** des listes simplement chaînées, introduit par les définitions suivantes :

```
type Cellule;

type Liste is access Cellule;

type Cellule is record
    Val : Integer;
    Suiv : Liste;
end record;

procedure Liberer is new Ada.Unchecked_Deallocation (Cellule, Liste);
```

FIG. 1 – Définition du type **Liste**

```

package FilePrio is

    subtype Elt is Integer range ...;

    function EstVide return Boolean;
    -- retourne True ssi la file ne contient aucun élément.

    procedure Vider;
    -- garantit EstVide = True

    procedure Inserer (X : in Elt);
    -- garantit une nouvelle occurrence de X est insérée dans la file.

    procedure DetruireMax (Max : out Elt);
    -- requiert EstVide = False
    -- garantit une occurrence de Max est retirée de la file,
    -- et Max est >= à tous les éléments restants dans la file.

end FilePrio;

```

FIG. 2 – Interface de FilePrio

En supposant Elt'First<=3 et 10<=Elt'Last,
on considère la séquence ci-contre.

A l'issue de cette séquence, la variable X vaut 10. La file de priorités contient ici les éléments suivants : "3" en deux exemplaires, "5" en quatre exemplaires, et "10" en un exemplaire.

Si on fait ensuite un nouvel appel à DetruireMax (X), la variable X vaut encore 10, mais la file ne contient plus l'élément 10. Le maximum de la file est donc 5 (en quatre exemplaires).

```

        X : Elt;
begin
    ...
    Vider;
    Inserer (3);
    Inserer (10);
    Inserer (5);
    Inserer (5);
    Inserer (10);
    Inserer (5);
    Inserer (3);
    Inserer (5);
    DetruireMax (X);

```

FIG. 3 – Exemple d'utilisation de FilePrio

L'implémentation par tableau utilise les deux variables globales :

```

NbOcc : array (Elt) of Natural;
MaxFile : Elt;

```

L'invariant satisfait par ces deux variables du paquetage est la conjonction des deux propriétés :

1. si NbOcc = (others => 0) alors MaxFile = Elt'First;
2. si NbOcc /= (others => 0) alors NbOcc (MaxFile) /= 0 et pour tout I : Elt tel que NbOcc (I) /= 0, on a I <= MaxFile.

FIG. 4 – Variables globales de FilePrio dans implém. par tableau

I - Files de priorités (12 points)

L'objectif de cet exercice est d'étudier différentes implémentations d'une structure de données appelée *file de priorités*. Les implémentations abordées ici sont relativement naïves. Des implémentations plus efficaces seront traitées au deuxième semestre.

On travaille ici avec un paquetage **FilePrio** dont l'interface est donnée figure 2 page 2. L'état interne de ce paquetage représente une suite ordonnée d'éléments, appelée *file de priorités*. Ces éléments sont du type **Elt**, un sous-type intervalle de **Integer**. L'ordre sur les éléments est donc l'ordre par défaut sur **Integer**. Le mode d'emploi du paquetage précise que l'état initial (au chargement du programme) n'est pas défini. Il faut donc commencer par appeler **Vider** avant d'utiliser le reste du paquetage. La figure 3 page 2 donne un exemple d'utilisation du paquetage. Dans la suite, on note N le cardinal de **Elt** (le nombre d'éléments de l'intervalle **Elt'Range**).

▷ **Question 1 (1 point) :**

Utiliser ce paquetage pour définir une procédure **Tri** qui ordonne les éléments d'un tableau **T** suivant l'ordre décroissant.

```
type Tab is array (Integer range <>) of Elt;
procedure Tri (T : in out Tab);
```

1 Implémentation par un tableau

Dans un premier temps, on suppose que N est relativement petit. On représente la file par un tableau **NbOcc** qui associe à chaque élément son nombre d'occurrences (ou d'exemplaires) dans la suite. Voir figure 4 page 2. Par exemple, à l'issue de la suite d'instructions donnée à la figure 3, on a :

```
NbOcc = (3 => 2, 5 => 4, 10 => 1, others => 0)
```

Pour rendre **DetruireMax** plus efficace, on calcule à l'avance le maximum courant de la file dans une variable globale appelée **MaxFile**. Typiquement pour le **NbOcc** de l'exemple précédent, **MaxFile=10**. Plus précisément, ces 2 variables doivent satisfaire l'invariant du paquetage donné figure 4 page 2 : si la file est vide, **MaxFile** vaut **Elt'First**, sinon il vaut l'élément maximum de la file.

▷ **Question 2 (4 points) :**

Implémenter les sous-programmes du paquetage en optimisant les temps de calculs autant que possible.

1. Écrire **Vider** (0,5 point) : cette procédure doit établir l'invariant.
2. Écrire **EstVide** (0,5 point) en utilisant l'invariant : le coût en temps de cette fonction est constant en fonction de N .
3. Écrire **Inserer** (1 point) : cette procédure doit préserver l'invariant.
4. Écrire **DetruireMax** (1,5 points) : cette procédure doit préserver l'invariant.
5. Calcul de coût (0,5 point) : indiquer le coût en temps du tri de la question 1 pour cette implémentation en fonction de N et de **T'Length** (la taille du tableau en entrée).

2 Deux implémentations par une liste chaînée

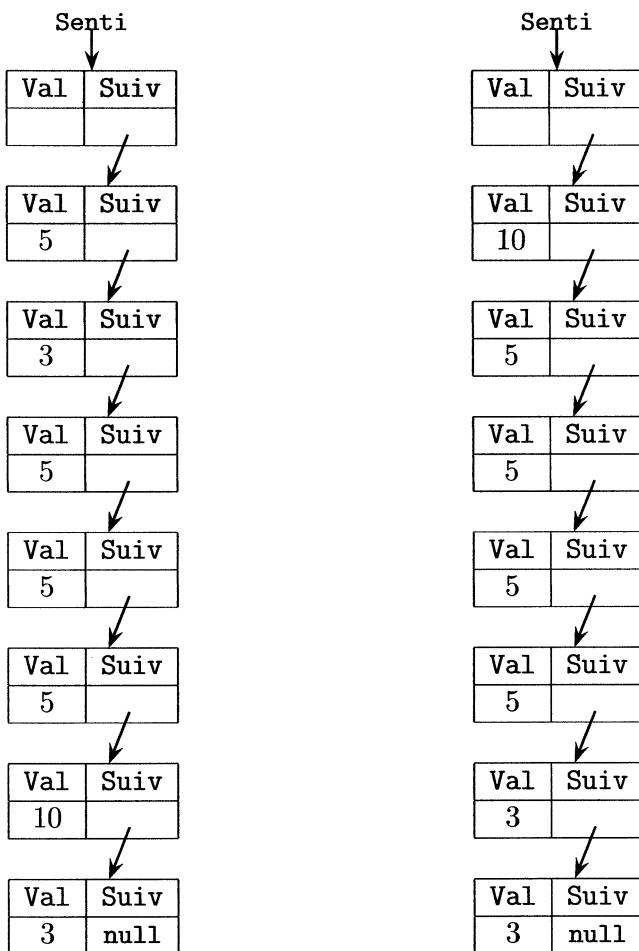
Maintenant, on étudie deux autres implémentations du paquetage, qui peuvent être intéressantes par rapport à l'implémentation précédente lorsque N est grand et que le nombre d'occurrences de chaque éléments de la file reste faible. Dans ces deux implémentations, le paquetage n'a qu'une seule variable globale interne qui est une liste simplement chaînée de type **Liste** (défini figure 1 page 1). Cette liste a une sentinelle de tête pour rendre les algorithmes réguliers. Dans votre code, vous devez utiliser cette sentinelle, et éviter des analyses par cas inutiles.

Cette sentinelle est allouée à l'initialisation du paquetage et elle n'est jamais libérée. Cette variable est donc simplement appelée **Senti** et déclarée comme ci-dessous dans le paquetage :

```
Senti : Liste := new Cellule;
```

Dans les deux implementations, l'invariant du paquetage garantit que **Senti** est une liste simplement chaînée **non vide** et bien formée (sans cycle et sans pointeur pendant), et que les éléments de la file sont ceux accessibles depuis la liste de tête **Senti.Suiv**.

Par exemple, les deux listes chaînées dessinées ci-dessous peuvent représenter la file obtenue à l'issue de la séquence donnée figure 3. Dans cette figure le champ **Val** de la cellule pointée par **Senti** est laissée vide pour signifier que sa valeur n'est jamais utilisée.



▷ **Question 3 (1 point) :**

Programmer `EstVide` et `Vider` en respectant le fait que `Senti` est une sentinelle de tête. la procédure `Vider` doit libérer les éventuelles cellules devenant inutilisées.

2.1 Première implémentation

Dans la première des 2 implémentations utilisant `Senti`, on ne constraint pas davantage l'invariant sur `Senti`.

▷ **Question 4 (1 point) :**

Écrire la procédure `Inserer` de sorte qu'elle place la nouvelle occurrence de `X` en tête de liste (dans le suivant de la sentinelle).

▷ **Question 5 (1 point) :**

Écrire une fonction interne au paquetage, appelée `CherchePredMax` qui a la spécification suivante :

```
function CherchePredMax return Liste is
  -- requiert la file contient au moins un élément.
  -- retourne un pointeur P non nul accessible depuis Senti
  -- tel que le suivant de P est non nul
  -- et la valeur de la cellule suivant P
  -- est le max des éléments de la file.
```

▷ **Question 6 (1 point) :**

Utiliser `CherchePredMax` pour écrire `DetruireMax`. La procédure `DetruireMax` doit libérer la cellule qui n'est plus utilisée.

2.2 Deuxième implémentation

Dans cette dernière implémentation, on constraint maintenant l'invariant sur `Senti` à garantir que la liste dont la tête est `Senti`. `Suiv` est triée par ordre décroissant sur les éléments (comme sur le dessin de droite, page 4).

▷ **Question 7 (2 points) :**

Reprogrammer `Inserer` et `DetruireMax` pour respecter ce nouvel invariant. La procédure `DetruireMax` doit libérer la cellule qui n'est plus utilisée.

▷ **Question 8 (1 point) :**

Indiquer les coûts en temps (en fonction de N et de `T'Length`) de la question 1 dans le meilleur cas et dans le pire cas pour chacune de vos 2 implémentations basées sur `Senti`. Dans cette analyse de coûts, on supposera qu'à l'appel de la procédure `Tri`, la file est déjà vide. On précisera quel est ce meilleur cas et ce pire cas (en fonction de `T`).

II - Énumération des permutations d'un mot (10 points)

Dans ce problème, on vous fournit la procédure `EnumV0` ci-dessous. Soit `S` une chaîne de caractères. Sous la précondition que *S est triée par ordre strictement croissant* (pour l'ordre par défaut sur les caractères), la procédure `EnumV0` affiche une fois et une seule chaque chaîne de taille `S'Length` constituée des caractères de `S`. Ces chaînes sont affichées par ordre lexicographique.

```
procedure EnumV0 (S : in String) is
    R : String (S'Range);
    procedure EnumRec (I : in Positive) is
        begin
            if I <= S'Last then
                for J in S'Range loop
                    R (I) := S (J);
                    EnumRec (I + 1);
                end loop;
            else
                Put_Line (R);
            end if;
        end EnumRec;
    begin
        EnumRec (S'First);
    end EnumV0;
```

▷ Question 9 (1,5 points) :

On s'intéresse à l'exécution de `EnumV0 ("ABC")`. Supposons qu'on en soit arrivé au point où `R = "BCC"`. Quel affichage produit alors un appel `EnumRec (S'First+1)`? Autrement dit, il faut ici écrire ce qu'affiche l'appel `EnumRec (S'First+1)` dans le contexte où `S = "ABC"` et `R = "BCC"`.

▷ Question 10 (1,5 points) :

Écrire une procédure `EnumV1` qui affiche la même chose que `EnumV0` mais en numérotant les lignes affichées par ordre croissant de 1 à $S'Length^{S'Length}$. Par exemple, si `S` vaut "AB", on obtient :

- 1 AA
- 2 AB
- 3 BA
- 4 BB

En s'inspirant de `EnumV0`, on va maintenant écrire une procédure `EnumV2` qui n'affiche que les `S'Length!` permutations de `S` (toujours sous la précondition que `S` est triée par ordre strictement croissant). Ces permutations doivent être affichées par ordre lexicographique. Par exemple, `EnumV2 ("ABC")` affiche le résultat suivant :

```
1 ABC
2 ACB
3 BAC
4 BCA
5 CAB
6 CBA
```

Il suffit pour cela de modifier `EnumRec` (`I`) de manière à ignorer les `J` tels que le caractère `S` (`J`) apparaît dans la tranche `R` (`S'First .. I - 1`). A cet effet, on introduit un tableau `AVisiter` externe à `EnumRec` qui indique les indices de `S` à visiter, c'est-à-dire ceux qu'il ne faut pas ignorer lors des appels récursifs à `EnumRec`.

▷ **Question 11 (3 points) :**

Compléter la procédure `EnumV2` ci-dessous :

```
procedure EnumV2 (S : in String) is

    R : String (S'Range);
    AVisiter : array (S'Range) of Boolean := (others => True);

    -- A COMPLETER EVENTUELLEMENT

    procedure EnumRec (I : in Positive) is
    begin
        -- A COMPLETER
    end EnumRec;

    -- A COMPLETER EVENTUELLEMENT
begin
    -- A COMPLETER
end EnumV2;
```

Indication : L'implémentation de `EnumRec` doit garantir que l'état final du tableau `AVisiter` est inchangé par rapport à son état initial. Autrement dit, `EnumRec` finit toujours par défaire les modifications qu'elle a effectuées en interne sur `AVisiter`. Cette technique fait que le coût en temps de `EnumV2` est similaire à celui de `EnumV0`.

Dans cette dernière question, on cherche à écrire une procédure `EnumV3` qui produise le même affichage que `EnumV2` mais avec une implémentation optimisée : il s'agit d'éviter de parcourir à chaque appel de `EnumRec` l'ensemble des indices de `S'Range` à la recherche des indices à visiter. Pour éviter ce parcours, on stocke l'ensemble des indices à visiter dans une liste simplement chaînée de type `Liste`, introduit à la figure 1 page 1. On remplace donc le tableau `AVisiter` précédent par une telle liste chaînée. Comme pour `AVisiter`, cette liste est laissée inchangée par `EnumRec` (même si elle est modifiée en interne).

Concrètement, au *début de la visite* d'un indice, on retire la cellule correspondante de la liste. Puis, à la *fin de la visite* de cet indice (après l'appel récursif à `EnumRec`), on remet la cellule à sa place initiale dans la liste. Pour faciliter ces 2 opérations (enlever une cellule de la liste et réinsérer la cellule dans la liste), on ajoute une sentinelle de tête dans la liste. Ces 2 opérations se font ainsi à coût constant sur la liste (voir figure 5).

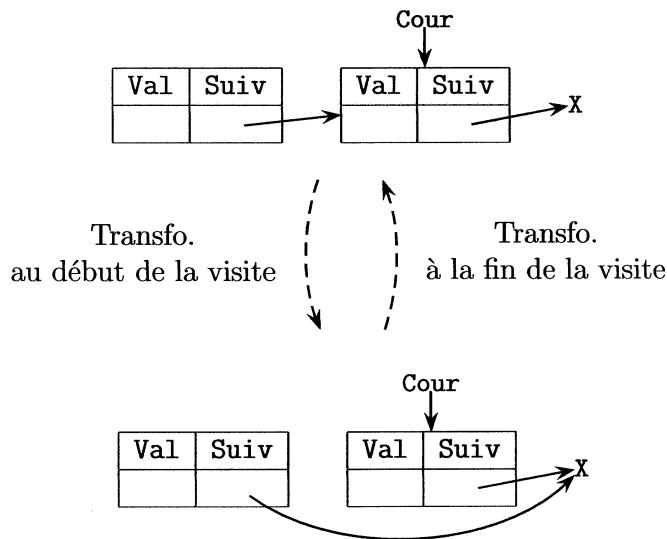


FIG. 5 – Transformations sur la liste lors de la visite de la cellule `Cour`

▷ **Question 12 (4 points) :**

Compléter la procédure `EnumV3` ci-dessous. On nomme `Senti` la variable qui pointe sur la sentinelle de tête dans la liste des indices à visiter.

```

procedure EnumV3 (S : in String) is

  R : String (S'Range);

  Senti : Liste;

  -- A COMPLETER EVENTUELLEMENT

  procedure EnumRec (I : in Positive) is
    Cour : Liste; -- cellule à visiter (cf. figure)
    -- A COMPLETER EVENTUELLEMENT
  begin
    -- A COMPLETER
  end EnumRec;

begin
  -- A COMPLETER
end EnumV3;

```

Examen à mi-parcours
Jeudi 27 novembre 2008 - 1h30

Documents manuscrits et calculatrice autorisés.

Le nom du groupe doit être noté en évidence sur la première page

NB :

- (i) Toutes les intégrales considérées dans cet énoncé le sont au sens de Lebesgue.
- (ii) Un soin particulier devra être apporté à la rédaction et à la justification des calculs.

Exercice 1 On définit, pour $x \geq 0$, la suite de fonctions :

$$f_n(x) = \left(1 + \frac{x}{n}\right)^n$$

1. Etudier la convergence simple de la suite (f_n) .

2. Calculer

$$\lim_{n \rightarrow +\infty} \int_0^n f_n(x) e^{-2x} dx$$

Exercice 2

1. Montrer que pour $x > 0$, on a :

$$\sum_{n=0}^{+\infty} e^{-nx} = \frac{1}{1 - e^{-x}}$$

2. Pour $p \geq 0$, on définit les intégrales:

$$I_p = \int_0^{+\infty} x^2 e^{-px} dx$$

Calculer I_p .

3. En utilisant les questions précédentes, calculer l'intégrale :

$$\int_0^{+\infty} \frac{x^2}{e^x - 1} dx$$

Exercice 3 On pose

$$F(t) = \int_0^{+\infty} \frac{\sin x}{x} e^{-xt} dx$$

1. Montrer que la fonction F est définie, continue et dérivable sur \mathbb{R}_+^* .
2. Montrer que pour tout $t > 0$, $F'(t) = -\frac{1}{1+t^2}$.
3. Calculer $\lim_{t \rightarrow +\infty} F(t)$.
4. Calculer l'expression de F sur \mathbb{R}_+^* à l'aide de fonctions élémentaires. En déduire que F se prolonge par continuité en 0.

Exercice 4 Soit $0 < a < b$. On considère dans \mathbb{R}^2 le domaine $\Omega =]0, +\infty[\times]a, b[$ et la fonction:

$$f(x, y) = e^{-xy}$$

1. Montrer que $f \in L^1(\Omega)$ et calculer $\iint_{\Omega} f(x, y) dx dy$.
2. En déduire la valeur de l'intégrale :

$$\int_0^{+\infty} \frac{e^{-ax} - e^{-bx}}{x} dx$$

Examen final
Mardi 27 janvier novembre 2009 - 1h30

Documents manuscrits, polycopié du cours et calculatrice autorisés.
Le nom du groupe doit être noté en évidence sur la première page

Exercice 1

1. Calculer les transformées de Fourier des fonctions suivantes:

$$f_0(x) = e^{-a|x|} \quad f_1(x) = \frac{x}{a^2+x^2} \quad f_2(x) = \frac{\sin(x)}{x}$$

2. Calculer l'intégrale suivante

$$\int_{\mathbb{R}} \frac{1}{(1+x^2)^2} dx$$

Exercice 2 On pose

$$f(x) = \int_0^\infty \frac{1}{\sqrt{t}} e^{-\frac{x^2}{2t} - \frac{t}{2}} dt$$

1. Montrer que $f \in L^1(\mathbb{R})$.
2. Calculer \hat{f} . En déduire que $f(x) = \sqrt{2\pi}e^{-|x|}$.

Exercice 3 Soit $a > 0$, $p \in \mathbb{N}$ et g une fonction C^∞ sur \mathbb{R} à support compact. On considère l'équation dans \mathbb{R} :

$$ay + (-1)^p y^{(2p)} = g$$

1. En utilisant la transformée de Fourier, trouver une solution $y \in C^\infty(\mathbb{R}) \cap L^1(\mathbb{R})$. On explicitera y sous forme intégrale.
2. Montrer que cette solution vérifie : $\|y\|_{L^\infty} \leq C\|y\|_{L^1}$, avec C constante.

Exercice 4

1. Pour $x \in \mathbb{R}$, on pose :

$$\rho(x) = \frac{1}{\pi} \frac{1}{1+x^2}, \quad \rho_n(x) = n\rho(nx) \text{ pour } n \geq 1$$

Calculer la limite dans $\mathcal{D}'(\mathbb{R})$ de la suite $T_n = T_{\rho_n}$.

2. Calculer dans $\mathcal{D}'(\mathbb{R})$ ($\alpha \neq 0$) :

$$\left(\frac{d^2}{dx^2} + \alpha^2 \right) Y(x) \sin(\alpha x)$$

où Y est la fonction de Heaviside $Y(x) = 1$ si $x \geq 0$ et 0 sinon.

(NB : la question revient à introduire la distribution-fonction $T = T_{Y(x) \sin(\alpha x)}$ et à calculer $\frac{d^2T}{dx^2} + \alpha^2 T$)

Exercice 1 On munit $E = C[0, 1]$ de la norme $\|\cdot\|_2$ ($\|u\|_2 = (\int_0^1 |u(t)|^2 dt)^{1/2}$ pour $u \in E$). Pour $u \in E$ et $x \in [0, 1]$ on pose:

$$T(u)(x) = \int_0^1 \max(x, t) u(t) dt.$$

Montrer que $T \in \mathcal{L}(E)$ et que $\|T\| \leq \sqrt{2}/2$.

Exercice 2 Soit $f : \mathbb{R}^N \rightarrow \mathbb{R}$ une application continue telle que

$$\lim_{\|x\| \rightarrow +\infty} f(x) = +\infty.$$

1) Montrer qu'il existe une constante $m \in \mathbb{R}$ telle que

$$f(x) \geq m \quad \forall x \in \mathbb{R}^N.$$

2) Montrer qu'il existe $y \in \mathbb{R}^N$ tel que

$$f(y) = \inf_{x \in \mathbb{R}^N} f(x).$$

Exercice 3 Soit $a > 0$ fixé. On munit $C[-a, a]$ de la norme $\|\cdot\|_\infty$ ($\|u\|_\infty = \sup_{-a \leq x \leq a} |u(x)|$ pour $u \in C[-a, a]$). Montrer que pour tout $v \in C[-a, a]$ il existe $u \in C[-a, a]$ unique tel que

$$u(x) = v(x) + \frac{1}{\pi} \int_{-a}^a \frac{u(y)}{1 + (x - y)^2} dy, \quad -a \leq x \leq a.$$

Exercice 4

Soit $E = C([0, 1], \mathbb{R})$ muni de la norme $\|\cdot\|_\infty$ ($\|u\|_\infty = \sup_{0 \leq x \leq 1} |u(x)|$ pour $u \in E$). Pour $u \in E$ et $x \in [0, 1]$ on pose:

$$J(u)(x) = u(0) + \int_0^x u(s)^2 ds.$$

Vérifier que $J(u) \in E$. Montrer que J est différentiable et donner la dérivée. Montrer que J est de classe C^1 .

Exercice 1 Soit H un espace de Hilbert sur $K = \mathbb{R}$ ou \mathbb{C} (on note $\langle ., . \rangle$ le produit scalaire). On suppose que H est muni d'une base hilbertienne $(e_n)_{n \in \mathbb{N}}$.

- 1) Montrer que $\langle x, y \rangle = \sum_{n=0}^{+\infty} \langle x, e_n \rangle \overline{\langle y, e_n \rangle}$ pour $x, y \in H$.
- 2) La suite $(e_n)_{n \in \mathbb{N}}$ est-elle convergente?

Exercice 2 On munit $E = C([-1, 1], \mathbb{R})$ du produit scalaire suivant

$$\langle u, v \rangle = \int_{-1}^1 u(t)v(t) dt, \quad u, v \in E,$$

et de la norme associée $\|.\|_2$. On définit

$$F = \{u \in E; u = 0 \text{ sur } [-1, 0]\}.$$

- 1) Montrer que F est un sous espace vectoriel fermé de E .
- 2) Déterminer F^\perp .
- 3) Montrer que $E \neq F \oplus F^\perp$. Conclure.

Exercice 3 On munit $E = \{u \in C^1([0, 1], \mathbb{R}); u(0) = 0\}$ de la norme $\|u\| = \max(\|u\|_\infty, \|u'\|_\infty)$ pour $u \in E$ et $F = C[0, 1]$ de la norme $\|.\|_\infty$. On définit $f : E \rightarrow F$ par

$$f(u) = u' + e^u, \quad u \in E.$$

- 1) Montrer que f est de classe C^1 et que $f'(0)$ est un isomorphisme de E sur F .
- 2) Montrer qu'il existe $a, b > 0$ tels que pour tout $h \in B(1, a) \subset F$ il existe $u \in B(0, b) \subset E$ unique vérifiant

$$u' + e^u = h.$$

Exercice 4 Pour $u \in E = \{u \in C^1([0, 1], \mathbb{R}); u(0) = 0\}$ on pose :

$$J(u) = \int_0^1 u'(s)^2 ds.$$

Montrer que J est strictement convexe sur E .

Architecture 1 :
Circuits Numériques et Eléments d'Architecture
Partiel de mi-semestre

ENSIMAG 1A

Année scolaire 2008–2009

- Durée : 2h. Tous documents et calculatrices autorisés.
- Le barème est donné à titre indicatif.
- Les exercices sont indépendants et peuvent être traités dans le désordre.

**VOUS DEVEZ RENDRE LES 3 EXERCICES SUR DES
COPIES SÉPARÉES**

Ex. 1 : Circuit comparateur d'entiers naturels (7 pts)

Dans cet exercice, on va construire un circuit combinatoire capable de comparer des entiers naturels. Ce circuit prend en entrée deux entiers naturels A et B codés sur 4 bits chacun, et produit trois sorties booléennes : $EQ = 1$ ssi les deux entiers sont égaux, $LT = 1$ ssi $A < B$ et $GT = 1$ ssi $A > B$.

Question 1 Construire la partie du circuit générant la sortie EQ valant 1 ssi $A = B$, en utilisant des portes logiques de base.

Question 2 Remplir une table de vérité partielle explicitant le calcul de la sortie LT , en fonction des bits $a(3), a(2), a(1)$ et $a(0)$ composant A , et $b(3), b(2), b(1)$ et $b(0)$ composant B . Indice : commencer par évaluer la sortie en partant des bits de poids fort des entrées, et progresser vers les bits de poids faibles.

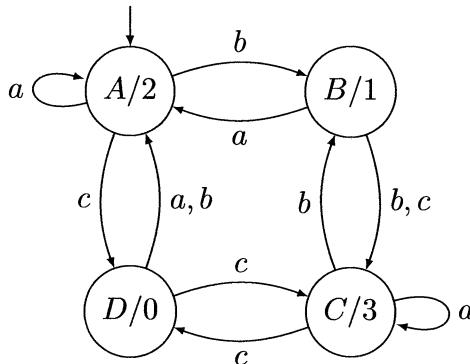
Question 3 En déduire une équation booléenne de la sortie LT . On ne demande pas de chercher à simplifier cette équation.

Question 4 Dessiner le schéma correspondant à l'équation précédente en utilisant des portes logiques de base.

Question 5 Comment peut-on ajouter simplement la sortie $GT = 1$ ssi $A > B$? On demande la solution la plus simple possible.

Ex. 2 : Synthèse d'automate (5 pts)

Soit l'automate de Moore suivant, qui modélise le fonctionnement d'un circuit prenant en entrée une séquence de caractères et produisant en sortie un entier :



Soient :

- $E = \{a, b, c\}$ le vocabulaire d'entrée ;
- $Q = \{A, B, C, D\}$ l'ensemble des états ;
- $S = \{0, 1, 2, 3\}$ le vocabulaire de sortie.

On impose le codage ci-dessous :

Entrée	e_1	e_0
a	0	0
b	0	1
c	1	0

État	q_1	q_0
A	0	0
B	0	1
C	1	0
D	1	1

Sortie	s_1	s_0
0	0	0
1	0	1
2	1	0
3	1	1

Question 1 Remplir la table des transitions à partir de l'automate de Moore :

Etat courant	Entrée	Etat suivant
A	a	A
...

Question 2 Donner les expressions simplifiées des transitions et des sorties de l'automate, en utilisant des tableaux de Karnaugh. Vous devez respecter le codage défini ci-dessus, dans le cas contraire votre réponse sera automatiquement comptée comme fausse.

Question 3 Dessiner le schéma du circuit en vous basant sur les équations trouvées à la question précédente. Vous utiliserez des bascules D, ainsi que les portes logiques dont vous aurez besoin. La clarté du schéma sera un critère déterminant dans la notation.

Question 4 Dessiner un chronogramme faisant apparaître les valeurs au cours du temps des signaux e_1 , e_0 , q_1 , q_0 , s_1 et s_0 pour la séquence « abcccba » en entrée. On considère que les entrées sont stabilisées bien avant le front montant de l'horloge.

Ex. 3 : Construction de circuits complexes (8 pts)

On travaille dans cet exercice sur un circuit implantant l'algorithme de la multiplication de deux entiers naturels vu en TD et TP d'algorithme :

```

procedure Mult is
    X, Y, R: Natural;
begin
    R := 0; Get(X); Get(Y);
    while Y /= 0 loop
        if Y(0) = 1 then -- Y(0) est le bit de poids faible de Y
            R := R + X;
        end if;
        X := X * 2; Y := Y / 2; -- division entière
    -- fin_iter
    end loop;
    Put(R);
end;

```

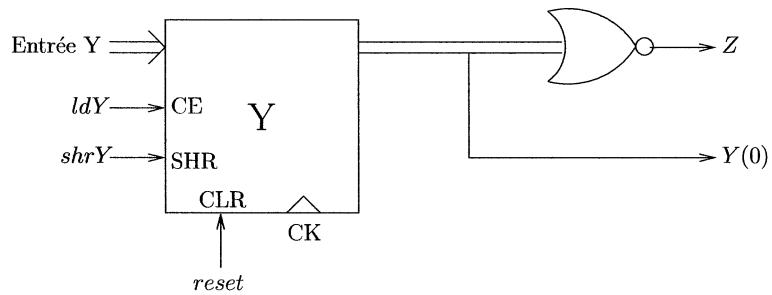
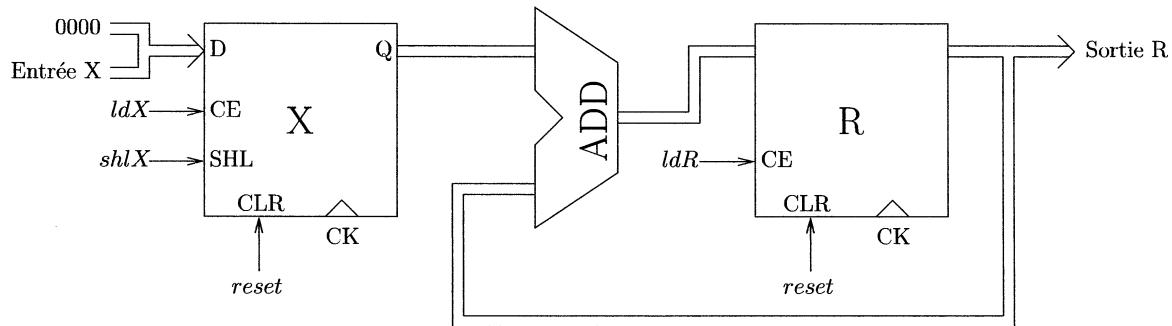
Le circuit comporte deux entrées sur 4 bits chacune, via lesquelles le circuit charge les valeurs initiales des paramètres X et Y (ce qui est symbolisé dans l'algorithme par les appels à la procédure `Get`) au début du calcul. Il dispose également d'une sortie sur 8 bits, par laquelle le circuit envoie le résultat de la multiplication à la fin du calcul (symbolisé dans l'algorithme par l'appel à la procédure `Put`).

Question 1 Dérouler à la main de fonctionnement de l'algorithme de la multiplication pour $X = 12_{10}$ et $Y = 5_{10}$, en remplissant le tableau ci-dessous avec les valeurs en binaire des variables au point `fin_iter`, jusqu'à ce que $Y = 0$:

X	Y	R
Init		
00001100_2	0101_2	00000000_2
Première itération		
...

Question 2 Justifier précisément pourquoi les variables X et R doivent être stockées dans des registres 8 bits.

Soit la partie opérative du circuit multiplicateur. La valeur initiale de X étant sur 4 bits, on l'étend sur 8 bits en forçant les 4 bits de poids fort à 0.



Question 3 A quels types de registres particuliers correspondent les registres X et Y dans la PO ci-dessus ? Donner leurs noms précis.

Question 4 Donner le graphe de la partie contrôle utilisant la PO ci-dessus et implantant l'algorithme de la multiplication. Vous prendrez soin de préciser les valeurs de toutes les sorties de la partie contrôle, correspond aux entrées de la PO. On ne se préoccupe pas de la synchronisation avec l'extérieur, et on considère que le circuit revient dans son état initial après avoir passé un cycle à émettre la valeur du résultat.

Architecture 1 :
Circuits Numériques et Eléments d'Architecture
Examen de 1^{re} session

ENSIMAG 1A

Année scolaire 2008–2009

- Durée : 3h. Tous documents et calculatrices autorisés.
- Le barème est donné à titre indicatif.
- Les exercices sont indépendants et peuvent être traités dans le désordre.
- Dans tout l'examen, les réponses **précises** et **concises** seront favorisées.

**VOUS DEVEZ RENDRE LES 3 EXERCICES SUR DES
COPIES SÉPARÉES**

Ex. 1 : Conception CMOS (5 pts)

Note : les questions 2, 3 et 4 (et partiellement la 5) de cet exercice peuvent être traitées même si vous n'avez réussi à déchiffrer le masque de la question 1.

Soit le masque donné en annexe, dont la légende est la suivante :

- les pistes hachurées de la gauche vers la droite (en bas) correspondent à du dopage $N+$;
- les pistes hachurées de la droite vers la gauche (en haut) correspondent à du dopage $P+$;
- les pistes quadrillées correspondent à du polysilicium ;
- le rectangle en trait pointillés est un caisson N ;
- les pistes blanches correspondent à du métal.

Question 1 Donner le schéma en transistors CMOS correspondant à ce masque.

Question 2 Dessiner le schéma CMOS correspondant à l'équation $d = \overline{a.b} + \overline{a}.c$ en utilisant la méthode par synthèse globale (*i.e.* PMOS en haut, NMOS en bas, comme en TD).

Question 3 Soit le circuit calculant $e = \overline{d}$. Donner la table de vérité de ce circuit. Quel est son nom ?

Question 4 En utilisant le théorème de De Morgan, exprimer l'équation de e uniquement en termes de NAND à 2 entrées et d'inverseurs. Dessiner le schéma CMOS du circuit correspondant à cette équation par composition des schémas CMOS du NAND et du NOT vus en TD.

Question 5 Comparer les 3 circuits CMOS dessinés dans cet exercice et conclure.

Ex. 2 : Programmation d'un micro-contrôleur (4 pts)

On va travailler dans cet exercice sur le micro-contrôleur PIC utilisé lors du TP5. Les documentations utiles se trouvent en annexe du sujet.

Question 1 Désassembler le code PIC ci-dessous (*i.e.* retrouver les instructions en assembleur PIC correspondant au code hexadécimal donné).

Hexadécimal	Assembleur
0x1683	
0x0185	
0x0181	
0x018B	
0x1283	

Hexadécimal	Assembleur
0x0191	
0x0A91	
0x0811	
0x0085	
0x2806	

Question 2 Que fait le programme ci-dessus ? On ne vous demande pas de détailler ce que fait chaque instruction mais d'expliquer le but général du programme (*e.g.* « ce programme calcule les 1024 premières décimales de π »).

Question 3 Expliquer le rôle de la première instruction du programme. Vous serez noté sur la **précision** de votre réponse.

Ex. 3 : Conception d'un processeur (11 pts)

On va travailler dans cet exercice sur le processeur dont on a étudié les parties contrôle et opérative lors des TD 11 et 12. On rappelle la PO en annexe. On considère que ce processeur est connecté à une mémoire asynchrone via un bus d'adresses sur 16 bits et un bus de données sur 8 bits, comme vu en TD.

On souhaite ajouter au processeur la gestion de trois instructions supplémentaires :

- MV #imm, SP : cette instruction copie dans le registre SP la valeur immédiate contenue dans l'instruction ;
- PUSH %ri : cette instruction commence par soustraire 1 à SP, puis elle stocke le contenu du registre %ri dans la case mémoire dont l'adresse est contenue dans SP (la valeur de SP prise en compte est celle après la décrémentation) ;
- POP %ri : cette instruction stocke dans le registre %ri le contenu de la case mémoire d'adresse SP puis ajoute 1 à SP.

On doit donc ajouter un nouveau registre à la PO : le registre SP¹ dont le rôle est de contenir l'adresse d'une zone de la mémoire² où on va ranger des données. On donne ci-dessous la table de vérité de ce registre :

Entrées							Sortie
RST	D(7:0)	LDH	LDL	INC	DEC	CK	SP(15:0)
1	-	-	-	-	-	-	$SP(15:0) = 0$
0	X(7:0)	1	0	-	-	↑	$SP(15:8) = X(7:0)$
0	X(7:0)	0	1	-	-	↑	$SP(7:0) = X(7:0)$
0	-	0	0	1	0	↑	$SP = SP_{prec} + 1$
0	-	0	0	0	1	↑	$SP = SP_{prec} - 1$

Ce registre a donc le fonctionnement suivant :

- si $RST = 1$, le registre est mis à 0 de façon asynchrone ;
- si $LDH = 1$ et $LDL = 0$, on charge dans les 8 bits de poids fort de SP la valeur en entrée du registre ;
- si $LDH = 0$ et $LDL = 1$, on charge dans les 8 bits de poids faible de SP la valeur en entrée du registre ;
- si $LDH = LDL = DEC = 0$ et $INC = 1$, on incrémente le contenu de SP ;
- si $LDH = LDL = INC = 0$ et $DEC = 1$, on décrémente le contenu de SP ;
- on ne doit pas avoir en même temps LDH et LDL (resp. INC et DEC) à 1.

Question 1 Comment peut-on très simplement relier la sortie de SP au bus adresse (indication : un seul composant de la PO à modifier) ?

Question 2 A quoi doit-on relier l'entrée $D(7:0)$ du registre SP pour pouvoir implanter l'instruction MV #imm, SP ? Justifier brièvement.

Dans les questions ci-dessous, on demande systématiquement la description RTL et la liste exhaustives des signaux de contrôle de la PO. On ne demande pas de rappeler les états du *fetch* ni de préciser le décodage des instructions.

Dans toutes les descriptions demandées, on considérera que tous les signaux de contrôle non-précisés valent 0, sauf les signaux de contrôle de la mémoire qui valent 1 par défaut. Vous donnerez systématiquement la description la plus parallélisée possible.

Question 3 Donner la description RTL de l'instruction MV #imm, SP, puis lister tous les signaux de contrôle associés aux différents états de l'automate. Vous utiliserez des signaux de contrôle portant le même nom que les entrées des registres (*e.g.* LDH, INC, ...). N'oubliez pas de préciser les signaux de contrôle de la mémoire !

Question 4 Donner la description RTL de l'instruction POP %ri, puis lister tous les signaux de contrôle associés aux états.

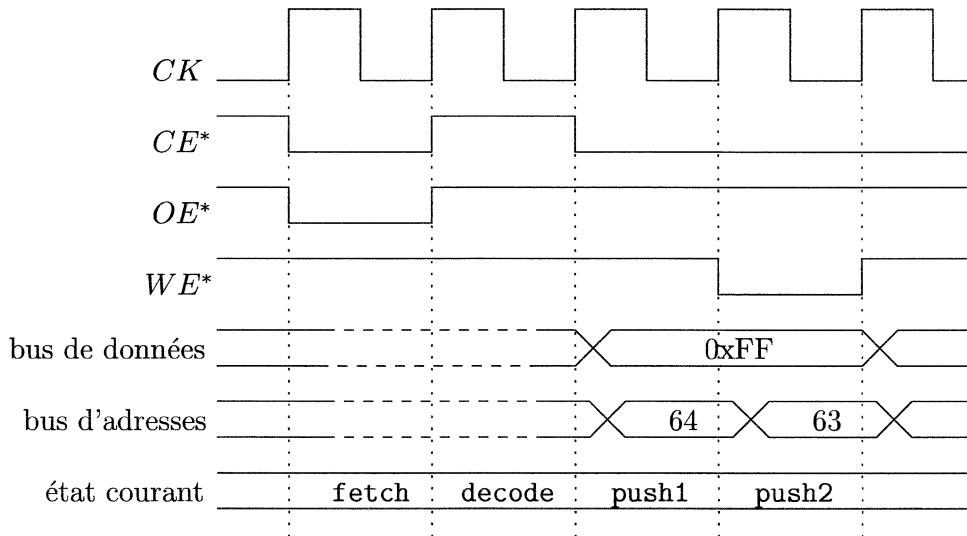
On va maintenant implanter l'instruction PUSH %ri. Dans une première version, on considère que l'écriture dans la mémoire peut se faire en 1 cycle sans problème de stabilisation.

¹connu sous le nom de « pointeur de pile » (*Stack Pointer*).

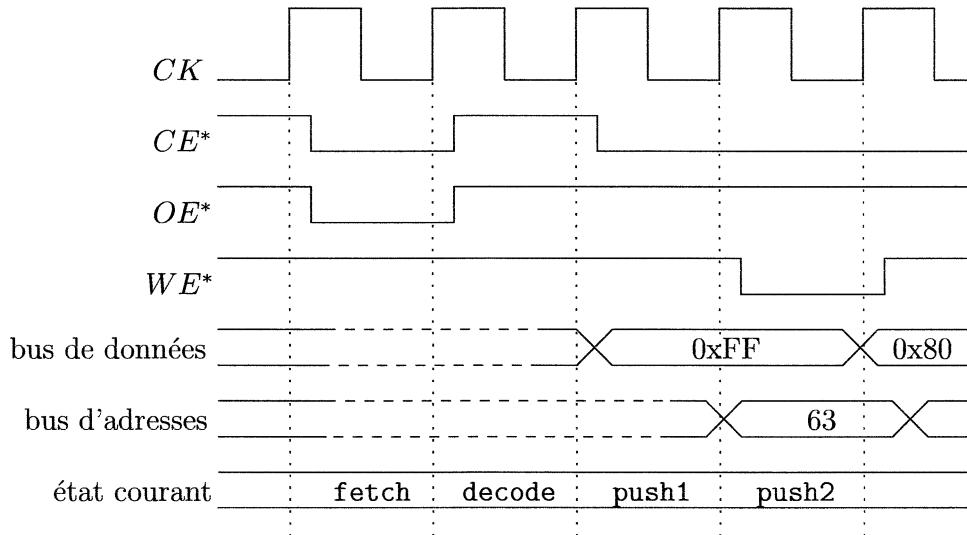
²en général appelée « pile » (*stack*).

Question 5 Donner la description RTL de l'instruction PUSH %ri, puis lister tous les signaux de contrôle associés aux états.

On considère les deux chronogrammes ci-dessous, correspondant à l'exécution de l'instruction PUSH %r0 avec SP valant initialement 64 et %r0 contenant la valeur -1 (ces deux chronogrammes correspondent à deux traces d'exécution théoriquement possibles de l'instruction).



Chronogramme 1.



Chronogramme 2.

Question 6 Expliquer précisément pourquoi dans l'état suivant push2, on trouve que :

- pour l'exécution correspondant au chronogramme 1, la valeur -1 est écrite aux adresses 63 et 64 ;
- pour l'exécution correspondant au chronogramme 2, la valeur -128 est écrite à l'adresse 63.

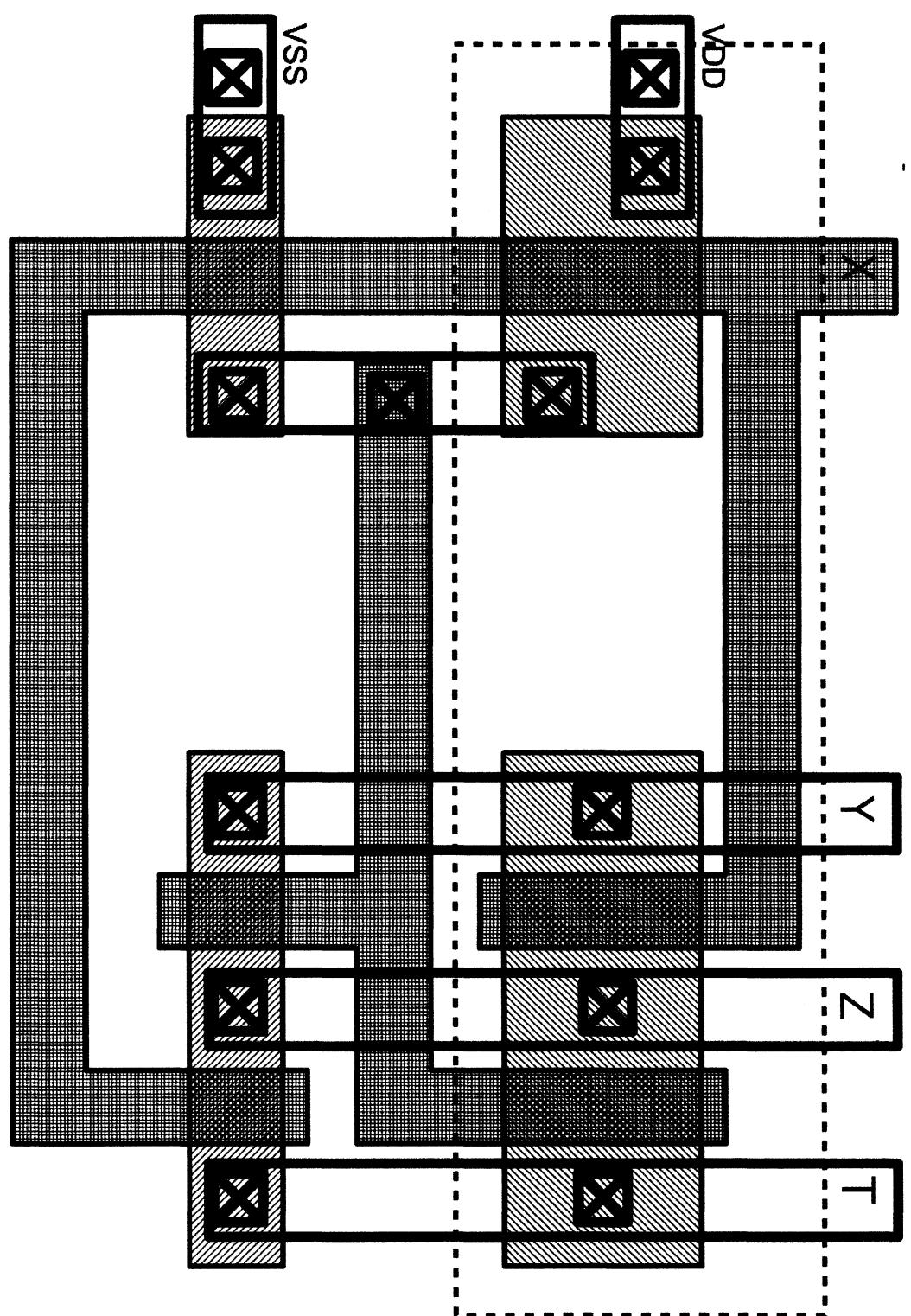
Question 7 Donner une description RTL de l'instruction PUSH `%ri` qui ne présente pas ces risques d'erreur, puis lister tous les signaux de contrôle associés aux états. On utilisera la notation `<-` pour symboliser la non-écriture mémoire en RTL.

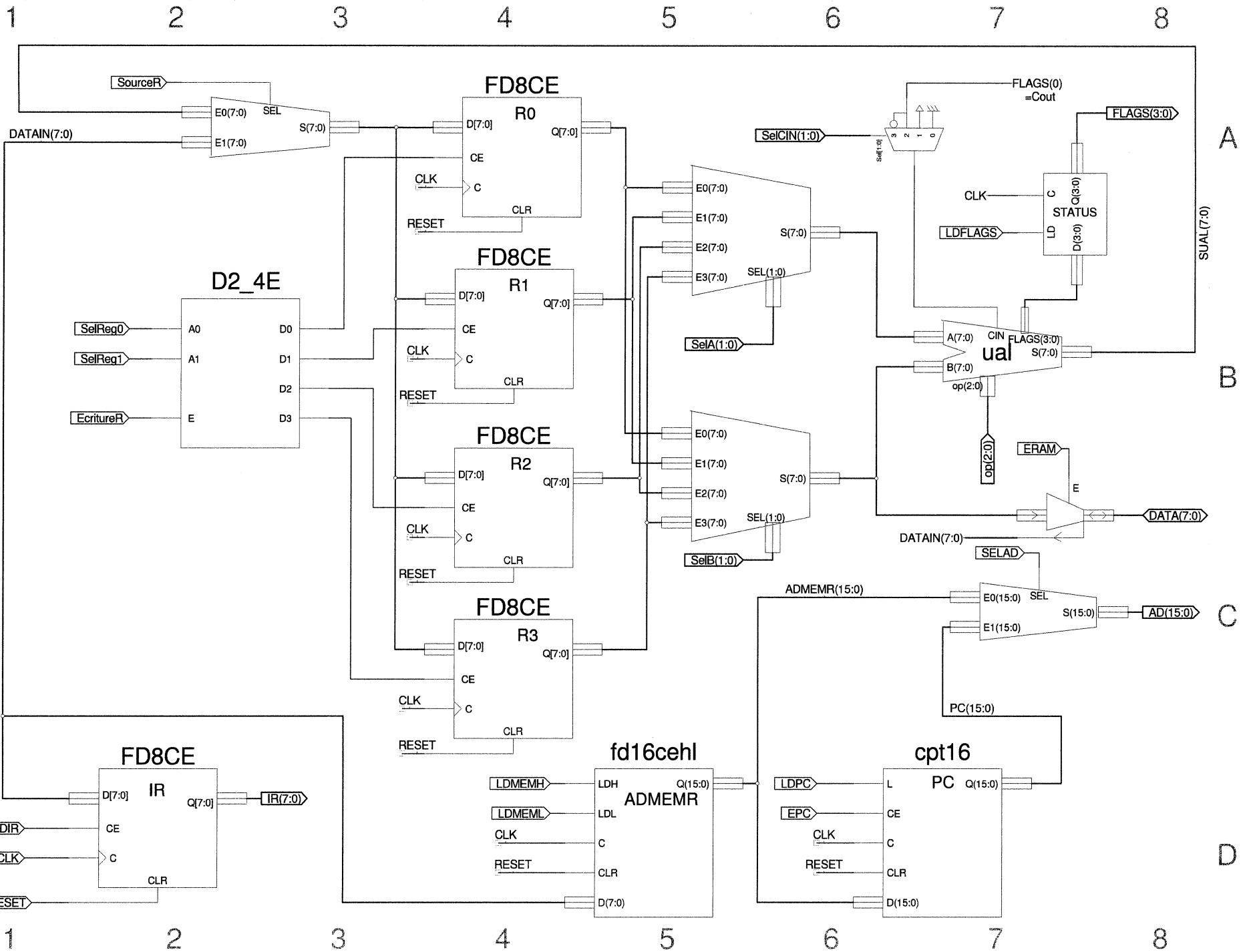
Les instructions PUSH et POP sont notamment utilisées pour passer des paramètres à des fonctions et procédures. Soit par exemple la procédure `Affiche(A, B, C: in Character)` qui affiche à l'écran les caractères (sur 8 bits chacun) passés en paramètre. Cette procédure s'attend à trouver ses paramètres dans la pile : le paramètre A est en sommet de pile, le paramètre B juste en dessous, et le paramètre C encore en dessous.

Question 8 En utilisant les instructions PUSH et POP, ainsi que toutes les instructions supportées par le processeur étudié dans les TD 11 et 12, écrire la séquence d'instructions correspondant à l'appel de la procédure `Affiche` avec pour paramètres (dans l'ordre) `'a'`, `'b'` et `'c'`. Autrement dit, vous devez écrire la séquence d'instructions correspondant aux étapes suivantes :

1. mettre les valeurs à passer en paramètre dans des registres ;
2. copier ces registres dans la pile ;
3. appeler la fonction `Affiche` ;
4. remettre la pile dans l'état où elle était avant l'étape 1.

Pour l'étape 3, vous écrirez simplement `CALL Affiche`.





PIC16F84A

2.0 MEMORY ORGANIZATION

There are two memory blocks in the PIC16F84A. These are the program memory and the data memory. Each block has its own bus, so that access to each block can occur during the same oscillator cycle.

The data memory can further be broken down into the general purpose RAM and the Special Function Registers (SFRs). The operation of the SFRs that control the "core" are described here. The SFRs used to control the peripheral modules are described in the section discussing each individual peripheral module.

The data memory area also contains the data EEPROM memory. This memory is not directly mapped into the data memory, but is indirectly mapped. That is, an indirect address pointer specifies the address of the data EEPROM memory to read/write. The 64 bytes of data EEPROM memory have the address range 0h-3Fh. More details on the EEPROM memory can be found in Section 3.0.

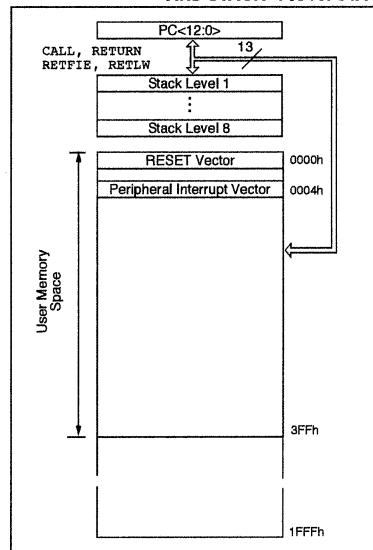
Additional information on device memory may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023).

2.1 Program Memory Organization

The PIC16FXX has a 13-bit program counter capable of addressing an 8K x 14 program memory space. For the PIC16F84A, the first 1K x 14 (0000h-03FFh) are physically implemented (Figure 2-1). Accessing a location above the physically implemented address will cause a wraparound. For example, for locations 20h, 420h, 820h, C20h, 1020h, 1420h, 1820h, and 1C20h, the instruction will be the same.

The RESET vector is at 0000h and the interrupt vector is at 0004h.

FIGURE 2-1: PROGRAM MEMORY MAP AND STACK - PIC16F84A



PIC16F84A

2.2 Data Memory Organization

The data memory is partitioned into two areas. The first is the Special Function Registers (SFR) area, while the second is the General Purpose Registers (GPR) area. The SFRs control the operation of the device.

Portions of data memory are banked. This is for both the SFR area and the GPR area. The GPR area is banked to allow greater than 116 bytes of general purpose RAM. The banked areas of the SFR are for the registers that control the peripheral functions. Banking requires the use of control bits for bank selection. These control bits are located in the STATUS Register. Figure 2-2 shows the data memory map organization.

Instructions MOVWF and MOVF can move values from the W register to any location in the register file ("F"), and vice-versa.

The entire data memory can be accessed either directly using the absolute address of each register file or indirectly through the File Select Register (FSR) (Section 2.5). Indirect addressing uses the present value of the RP0 bit for access into the banked areas of data memory.

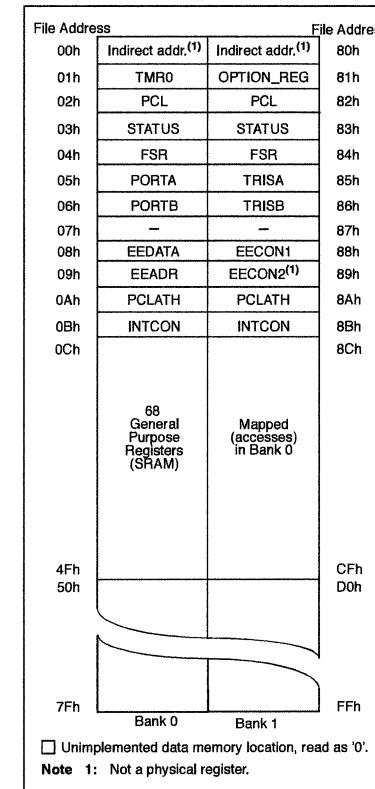
Data memory is partitioned into two banks which contain the general purpose registers and the special function registers. Bank 0 is selected by clearing the RP0 bit (STATUS<5>). Setting the RP0 bit selects Bank 1. Each Bank extends up to 7Fh (128 bytes). The first twelve locations of each Bank are reserved for the Special Function Registers. The remainder are General Purpose Registers, implemented as static RAM.

2.2.1 GENERAL PURPOSE REGISTER FILE

Each General Purpose Register (GPR) is 8-bits wide and is accessed either directly or indirectly through the FSR (Section 2.5).

The GPR addresses in Bank 1 are mapped to addresses in Bank 0. As an example, addressing location 0Ch or 8Ch will access the same GPR.

FIGURE 2-2: REGISTER FILE MAP - PIC16F84A



PIC16F84A

2.3 Special Function Registers

The Special Function Registers (Figure 2-2 and Table 2-1) are used by the CPU and Peripheral functions to control the device operation. These registers are static RAM.

TABLE 2-1: SPECIAL FUNCTION REGISTER FILE SUMMARY

Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on RESET	Details on page			
Bank 0														
00h	INDF	Uses contents of FSR to address Data Memory (not a physical register)						----	----	11				
01h	TMR0	8-bit Real-Time Clock/Counter						xxxx xxxx	xxxx xxxx	20				
02h	PCL	Low Order 8 bits of the Program Counter (PC)						0000 0000	0000 0000	11				
03h	STATUS ⁽²⁾	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	8			
04h	FSR	Indirect Data Memory Address Pointer 0						xxxx xxxx	xxxx xxxx	11				
05h	PORTA ⁽⁴⁾	-	-	-	RA4/TOCKI	RA3	RA2	RA1	RA0	---x xxxx	16			
06h	PORTB ⁽⁵⁾	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	16			
07h	-	Unimplemented location, read as '0'						-	-					
08h	EEDATA	EEPROM Data Register						xxxx xxxx	xxxx xxxx	13,14				
09h	EEADR	EEPROM Address Register						xxxx xxxx	xxxx xxxx	13,14				
0Ah	PCLATH	-	-	-	Write Buffer for upper 5 bits of the PC ⁽¹⁾			---0 0000	0000 0000	11				
0Bh	INTCON	GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	10			
Bank 1														
80h	INDF	Uses Contents of FSR to address Data Memory (not a physical register)						----	----	11				
81h	OPTION_REG	RBPU	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0	1111 1111	9			
82h	PCL	Low order 8 bits of Program Counter (PC)						0000 0000	0000 0000	11				
83h	STATUS ⁽²⁾	IRP	RP1	RP0	TO	PD	Z	DC	C	0001 1xxx	8			
84h	FSR	Indirect data memory address pointer 0						xxxx xxxx	xxxx xxxx	11				
85h	TRISA	-	-	-	PORTA Data Direction Register			--1 1111	1111	16				
86h	TRISB	PORTB Data Direction Register						1111 1111	1111 1111	18				
87h	-	Unimplemented location, read as '0'						-	-					
88h	EECON1	-	-	-	EEIF	WRERR	WREN	WR	RD	---0 x000	13			
89h	EECON2	EEPROM Control Register 2 (not a physical register)						---	---	14				
0Ah	PCLATH	-	-	-	Write buffer for upper 5 bits of the PC ⁽¹⁾			---0 0000	0000 0000	11				
0Bh	INTCON	GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF	0000 000x	10			

Legend: x = unknown, u = unchanged. - = unimplemented, read as '0', q = value depends on condition

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a slave register for PC<12:8>. The contents of PCLATH can be transferred to the upper byte of the program counter, but the contents of PC<12:8> are never transferred to PCLATH.

2: The TO and PD status bits in the STATUS register are not affected by a MCLR Reset.

3: Other (non power-up) RESETS include: external RESET through MCLR and the Watchdog Timer Reset.

4: On any device RESET, these pins are configured as inputs.

5: This is the value that will be in the port output latch.

PIC16F84A

2.3.1 STATUS REGISTER

The STATUS register contains the arithmetic status of the ALU, the RESET status and the bank select bit for data memory.

As with any register, the STATUS register can be the destination for any instruction. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to device logic. Furthermore, the TO and PD bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, CLR STATUS will clear the upper three bits and set the Z bit. This leaves the STATUS register as 000u uuu (where u = unchanged).

Only the BCF, BSF, SWAPF and MOVWF instructions should be used to alter the STATUS register (Table 7-2), because these instructions do not affect any status bit.

Note 1: The IRP and RP1 bits (STATUS<7:6>) are not used by the PIC16F84A and should be programmed as cleared. Use of these bits as general purpose R/W bits is NOT recommended, since this may affect upward compatibility with future products.

2: The C and DC bits operate as a borrow and digit borrow out bit, respectively, in subtraction. See the SUBLW and SUBWF instructions for examples.

3: When the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. The specified bit(s) will be updated according to device logic

REGISTER 2-1: STATUS REGISTER (ADDRESS 03h, 83h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	TO	PD	Z	DC	C

bit 7

bit 7-6 **Unimplemented:** Maintain as '0'

bit 5 **RP0:** Register Bank Select bits (used for direct addressing)

01 = Bank 1 (80h - FFh)
00 = Bank 0 (00h - 7Fh)

bit 4 **TO:** Time-out bit

1 = After power-up, CLRWDAT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3 **PD:** Power-down bit

1 = After power-up or by the CLRWDAT instruction
0 = By execution of the SLEEP instruction

bit 2 **Z:** Zero bit

1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit carry/borrow bit (ADDWF, ADDLW, SUBLW, SUBWF instructions) (for borrow, the polarity is reversed)

1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0 **C:** Carry/borrow bit (ADDWF, ADDLW, SUBLW, SUBWF instructions) (for borrow, the polarity is reversed)

1 = A carry-out from the Most Significant bit of the result occurred

0 = No carry-out from the Most Significant bit of the result occurred

Note: A subtraction is executed by adding the two's complement of the second operand.
For rotate (RRF, RLF) instructions, this bit is loaded with either the high or low order bit of the source register.

Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

-n = Value at POR

'1' = Bit is set

'0' = Bit is cleared

x = Bit is unknown

PIC16F84A

7.0 INSTRUCTION SET SUMMARY

Each PIC16CXX instruction is a 14-bit word, divided into an OPCODE which specifies the instruction type and one or more operands which further specify the operation of the instruction. The PIC16CXX instruction set summary in Table 7-2 lists **byte-oriented**, **bit-oriented**, and **literal and control** operations. Table 7-1 shows the opcode field descriptions.

For **byte-oriented** instructions, 'f' represents a file register designator and 'd' represents a destination designator. The file register designator specifies which file register is to be used by the instruction.

The destination designator specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the W register. If 'd' is one, the result is placed in the file register specified in the instruction.

For **bit-oriented** instructions, 'b' represents a bit field designator which selects the number of the bit affected by the operation, while 'f' represents the address of the file in which the bit is located.

For **literal and control** operations, 'k' represents an eight or eleven bit constant or literal value.

TABLE 7-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
w	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1) The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1
PC	Program Counter
TO	Time-out bit
PD	Power-down bit

The instruction set is highly orthogonal and is grouped into three basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal and control** operations

All instructions are executed within one single instruction cycle, unless a conditional test is true or the program counter is changed as a result of an instruction. In this case, the execution takes two instruction cycles with the second cycle executed as a NOP. One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μ s. If a conditional test is true or the program counter is changed as a result of an instruction, the instruction execution time is 2 μ s.

Table 7-2 lists the instructions recognized by the MPASM™ Assembler.

Figure 7-1 shows the general formats that the instructions can have.

Note: To maintain upward compatibility with future PIC16CXX products, do not use the OPTION and TRIS instructions.

All examples use the following format to represent a hexadecimal number:

0xhh

where hh signifies a hexadecimal digit.

FIGURE 7-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations			
13	8	7	6 0
OPCODE	d	f (FILE #)	
			d = 0 for destination W d = 1 for destination f f = 7-bit file register address
Bit-oriented file register operations			
13	10	9	7 6 0
OPCODE	b (BIT #)	f (FILE #)	
			b = 3-bit bit address f = 7-bit file register address
Literal and control operations			
General	8	7	0
OPCODE		k (literal)	
			k = 8-bit immediate value
CALL and GOTO instructions only			
13	11	10	0
OPCODE		k (literal)	
			k = 11-bit immediate value

A description of each instruction is available in the PICmicro™ Mid-Range Reference Manual (DS33023).

PIC16F84A

TABLE 7-2: PIC16CXXX INSTRUCTION SET

Mnemonic, Operands	Description	Cycles	14-Bit Opcode		Status Affected	Notes
			MSb	Lsb		
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF f, d	Add W and f	1	00	0111 dfff ffff	C,DC,Z	1,2
ANDWF f, d	AND W with f	1	00	0101 dfff ffff	Z	1,2
CLRF f	Clear f	1	00	0001 1fff ffff	Z	2
CLRW -	Clear W	1	00	0001 0xxx xxxx	Z	
COMF f, d	Complement f	1	00	1001 dfff ffff	Z	1,2
DECf f, d	Decrement f	1	00	0011 dfff ffff	Z	1,2
DECFSZ f, d	Decrement f, Skip if 0	1 (2)	00	1011 dfff ffff	Z	1,2,3
INCf f, d	Increment f	1	00	1010 dfff ffff	Z	1,2
INCFSZ f, d	Increment f, Skip if 0	1 (2)	00	1111 dfff ffff	Z	1,2,3
IORWF f, d	Inclusive OR W with f	1	00	0100 dfff ffff	Z	1,2
MOVF f, d	Move f	1	00	1000 dfff ffff	Z	1,2
MOVWF f	Move W to f	1	00	0000 1fff ffff	Z	1,2
NOP -	No Operation	1	00	0000 0xx0 0000		
RLF f, d	Rotate Left f through Carry	1	00	1101 dfff ffff	C	1,2
RRF f, d	Rotate Right f through Carry	1	00	1100 dfff ffff	C	1,2
SUBWF f, d	Subtract W from f	1	00	0010 dfff ffff	C,DC,Z	1,2
SWAPF f, d	Swap nibbles in f	1	00	1110 dfff ffff	Z	1,2
XORWF f, d	Exclusive OR W with f	1	00	0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF f, b	Bit Clear f	1	01	00bb bfff ffff		1,2
BSF f, b	Bit Set f	1	01	01bb bfff ffff		1,2
BTFSZ f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb bfff ffff		3
BTFSZ f, b	Bit Test f, Skip if Set	1 (2)	01	11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW k	Add literal and W	1	11	111x kkkk kkkk	C,DC,Z	
ANDLW k	AND literal with W	1	11	1001 kkkk kkkk	Z	
CALL -	Call subroutine	2	10	0kkk kkkk kkkk		TO,PD
CLRWDAT	Clear Watchdog Timer	1	00	0000 0110 0100		
GOTO k	Go to address	2	10	1kkk kkkk kkkk		
IORLW k	Inclusive OR literal with W	1	11	1000 kkkk kkkk	Z	
MOVLW k	Move literal to W	1	11	00xx kkkk kkkk		
RETIE -	Return from interrupt	2	00	0000 0000 1001		
RETlw k	Return with literal in W	2	11	01xx kkkk kkkk		
RETURN -	Return from Subroutine	2	00	0000 0000 1000		
SLEEP -	Go into standby mode	1	00	0000 0110 0011		TO,PD
SUBLW k	Subtract W from literal	1	11	110x kkkk kkkk	C,DC,Z	
XORLW k	Exclusive OR literal with W	1	11	1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., MOVF PORTB, 1), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

Note 2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 Module.

Note 3: If Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

Note: Additional information on the mid-range instruction set is available in the PICmicro™ Mid-Range MCU Family Reference Manual (DS33023).

ECONOMIE

DS de 29 janvier 2009

Durée : 2 heures

Tous documents autorisés sauf mobiles et ordinateurs portables

Vous traiterez, au choix, l'un des deux sujets suivants.

Sujet n°1 : Travail sur le texte ci-joint.

*Votre devoir devra suivre le plan de travail proposé.
L'ensemble ne doit pas dépasser quatre pages, soit une copie double.*

1- Synthèse

- Thème général de l'article
 - Question soulevée par l'auteur du texte
 - Réponse
 - Arguments principaux
- 2- Lister les **éléments du texte** que l'auteur ne développe pas et qu'il serait nécessaire d'expliquer pour une bonne compréhension de l'article : des notions, des mécanismes, des raisonnements, voire des références historiques.
- 3- Choisir, dans 2), **deux éléments** qui vous paraissent particulièrement importants et **rédiger**, pour chacun d'eux, **l'explication** qui vous paraît nécessaire à sa bonne compréhension. (Exclure de ce choix la seule définition de notions).
- 4- **Conclusion** : En quelques phrases, présentez ce qui vous paraît constituer les points forts ainsi que les limites éventuelles de l'article.

La BCE abaisse nettement son principal taux directeur à 2%

AFP – 15 janvier 2009

La Banque centrale européenne (BCE) a annoncé jeudi une baisse d'un demi point à 2% de son principal taux d'intérêt directeur, répondant comme l'attendait une majorité d'économistes aux récentes nouvelles calamiteuses sur le front de l'économie en zone euro.

Le principal taux est ramené ainsi à son plancher historique: il avait stationné à ce niveau entre juin 2003 et décembre 2005. En dix ans d'existence, la BCE n'est jamais descendue plus bas.

Et il s'agit de la quatrième baisses d'affilée, du jamais vu dans l'histoire de l'institution.

Le président de la Banque centrale européenne (BCE) Jean-Claude Trichet a estimé qu'il faudra attendre le "rendez-vous important" de la prochaine réunion de la BCE début mars pour une éventuelle nouvelle baisse de taux directeurs.

Interrogé lors d'une conférence de presse sur d'éventuelles nouvelles réductions des taux, le Français a souligné que la réunion de début mars "*était le prochain rendez-vous important*" en matière de politique monétaire, et exclu tout mouvement en février.

Cette déclaration est une façon pour le Français d'ouvrir la voie à une nouvelle baisse de taux en mars. La BCE publiera à cette occasion ses nouvelles prévisions de croissance et d'inflation.

Le Français va sans doute rester prudent pour ce qui est de la suite des événements monétaire, mais "*étant donné les nouvelles exécrables annoncées récemment pour l'économie de la zone euro, d'autres réductions de taux paraissent inévitables*", a régi Ben May, de Capital Economics.

"Nous croyons que les données vont être si mauvaises que la BCE sera forcée d'aller de 2% à une nouveau plancher historique de 1,5% en mars", estime de son côté Holger Schmieding, chef économiste pour l'Europe de la Bank of America.

Un léger suspense avait pourtant plané sur l'issue de la réunion. La BCE avait plutôt signalé son intention de faire une pause en janvier, après trois baisses successives entre octobre et décembre.

Mais les nouvelles économiques en zone euro sont alarmantes depuis le début de l'année, avec des statistiques régulièrement plus mauvaises que prévu, faisant craindre une récession plus grave qu'envisagé.

La confiance des chefs d'entreprises, affectée par le ralentissement de la demande intérieure et mondiale, et des consommateurs s'est effondrée en décembre, à un niveau jamais atteint. Le taux de chômage, avec 7,8% en novembre, n'avait pas été aussi élevé depuis 2006.

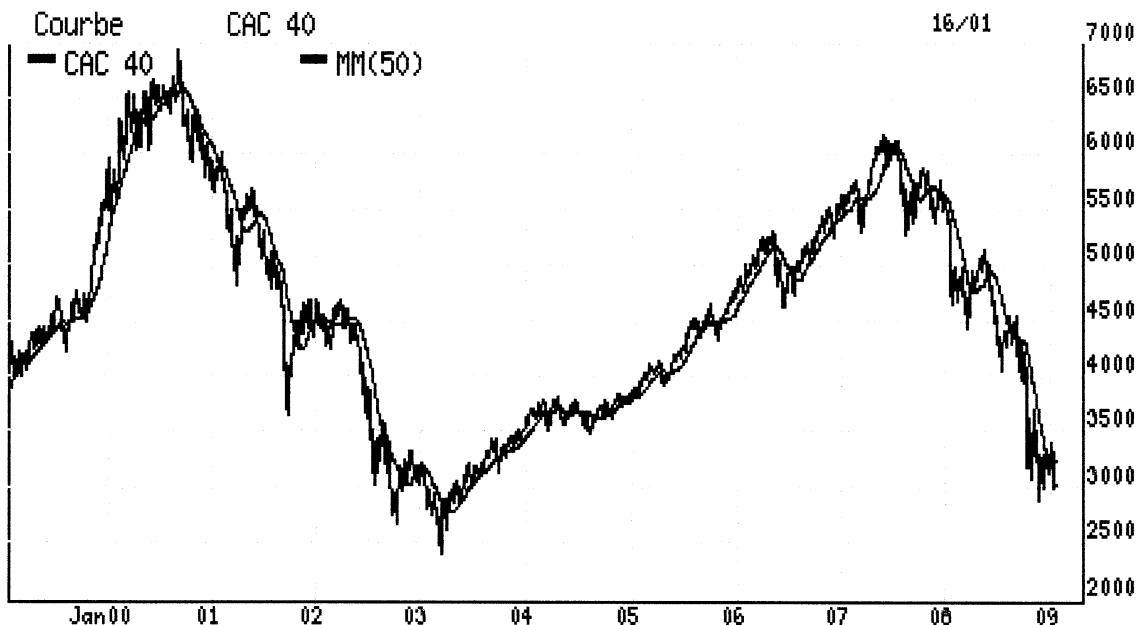
"Le pire est à venir" pour l'économie de la zone euro, avait d'ailleurs estimé mercredi le secrétaire général de l'Organisation pour la coopération et le développement économique (OCDE), Angel Gurria, ajoutant qu'il s'attendait en conséquence à une poursuite des baisses de taux.

L'inflation, contre laquelle la BCE doit lutter en priorité, n'est de son côté plus un souci. En décembre, elle est descendue à un taux de 1,6% sur un an, a confirmé jeudi l'office statistique européen Eurostat.

La puissante fédération des banques allemandes a salué le geste de la BCE grâce auquel elle "*apporte une contribution importante au succès des paquets de relance économique*" décidés dans les différents pays de la zone euro.

Sujet n°2

Evolution du CAC 40 (Historique : décembre 1999 – janvier 2009)



Copyright Boursorama - www.boursorama.com

Commentez la courbe du CAC 40.

Le commentaire est libre.

Veillez cependant à le structurer de façon classique (introduction, développements, conclusion) et à soigner la rédaction.

L'ensemble ne devra pas dépasser 6 pages.

ECONOMIE

Session de rattrapage - Juin-Juillet 2009

Durée : 2 heures

Tous documents autorisés - Mobiles et ordinateurs portables interdits.

Travail demandé : Sur la base du **texte ci-joint**, vous rédigerez, **au choix** :

- Un **commentaire libre**

ou

- Un commentaire adoptant le **plan suivant** :

1)- Lister les éléments du texte que l'auteur ne développe pas et qu'il serait nécessaire d'expliquer pour une bonne compréhension de l'article : des notions, des mécanismes, des raisonnements, voire des références historiques.

2)- Choisir, dans vos réponses à la question 1, deux éléments qui vous paraissent particulièrement importants et rédiger, pour chacun d'eux, l'explication qui vous paraît nécessaire à sa bonne compréhension. (Exclure de ce choix la seule définition de notions).

3)- Conclusion : En quelques phrases, présentez ce qui vous paraît constituer les principaux enjeux de l'article.

La BCE veut débloquer le crédit aux entreprises et aux ménages

Le Monde - Agence France Presse - 7 juin 2009

La Banque centrale européenne (BCE) a décidé, jeudi 7 mai, d'abaisser son principal taux d'intérêt directeur d'un quart de point au niveau inédit de 1 %. Il s'agit de la septième baisse depuis octobre, quand le taux de refinancement (le baromètre du crédit dans la zone euro) plafonnait encore à 4,25 %.

Le taux est ainsi à un « niveau approprié » mais pourrait être encore abaissé en théorie, a estimé Jean-Claude Trichet, le président de la BCE. Il a souligné la fragilité de l'économie en zone euro, qui s'est détériorée au premier trimestre « *nettement plus que prévu en mars* », quand la BCE a présenté ses prévisions économiques. En conséquence, ces dernières devraient être révisées « *significativement* » à la baisse en juin, et s'aligner sur celles de la Commission européenne notamment, qui a récemment prédit une chute de 4 % du produit intérieur brut de la zone euro en 2009.

Mais, au-delà de cette modification des taux d'intérêt, la BCE va surtout accorder aux banques des financements à un an, contre six mois jusqu'à présent. « *La BCE cherche ainsi à débloquer les rouages du financement des agents économiques* », entreprises et ménages, explique Cécile Prudhomme, du service Economie du *Monde*.

Introduction aux réseaux et aux télécom

Examen du 26/1/2008. Durée 2h00. Documents de cours, TD et TP autorisés.
Les trois grandes parties (GSM, Hotspot, écoute en amphi) sont indépendantes. Le barème est indicatif.

Couverture géographique du GSM et capacité de service (6 points)

En zone périurbaine, la densité de population peut approcher 500 habitants/km² (rapport du sénateur G. Larcher 1997/98). Pour de telles zones d'habitation, la pointe de trafic téléphonique se situe le soir entre 17h30 et 21h, avec une intensité moyenne de 15 mE par mobile.

Question 1. *En supposant que le taux d'abonnement de la population française est de 88,7% (chiffres Arcep pour septembre 2008), calculez l'intensité moyenne de trafic téléphonique par km² dans les zones périurbaines.*

Pour couvrir de telles zones, un opérateur mobile déploie des cellules hexagonales de rayon R.

Question 2. *En supposant que chaque cellule peut transporter 7 communications simultanées, calculez le rayon moyen de telles cellules*

Pour réaliser une telle couverture radio, l'opérateur choisit d'avoir une distance entre cellules de même fréquence d'au moins 5 km.

Question 3. *Expliquez l'intérêt d'avoir une distance minimale entre cellules de même fréquence.*

Question 4. *Trouvez un pas (i,j) de réutilisation de fréquence permettant d'obtenir au moins la distance minimale vue en Question 3. À quelle taille de cluster cela correspond-il ?*

Hotspot Wifi à péage

Présentation du service

De nombreux opérateurs installent des services d'accès sans fil par Wifi dans des lieux publics, tels que les hôtels, gares et aéroports. On appelle communément hotspot un tel service correspondant à un réseau local sans fil dans une zone restreinte à un bâtiment. On s'intéresse ici à un service payant mis en place par un opérateur au nom coloré de 'Marron'.

Le service vu de l'utilisateur doté d'un ordinateur portable est le suivant. Il détecte un réseau sans fil annoncé sous le nom : 'Marron_wifi'. Il indique à son ordinateur d'utiliser cette connexion de réseau sans fil. Il ne peut pas consulter sa messagerie ou faire d'autres accès sur Internet tant qu'il n'a pas lancé un navigateur accédant à

l'Internet. Son navigateur affiche alors non pas la page demandée (par ex sa page d'accueil habituelle) mais la page d'accueil d'un portail 'Wifi payant' de Marron. Il doit acheter un temps de communication (typiquement une ou plusieurs heures) sur ce site. À partir du moment où il a payé, le réseau Marron_wifi le connecte alors à l'Internet et il peut librement accéder à tous les services d'Internet.

Votre professeur de réseau, Roland Groz, se connecte depuis un ordinateur ordi-groz dans une gare. L'adresse IP de ordi-groz obtenue au démarrage de la connexion Wifi est 172.18.249.232. Il obtient aussi un serveur DNS qui est la machine hotspot_Marron_LaGare (nom canonique DNS, alias hsp_lg), d'adresse IP 172.18.249.193 sur le réseau interne (côté hotspot), et d'adresse IP publique 80.10.45.12 sur le réseau Internet. Cette même machine route également les communications entre la zone Wifi et l'Internet.

En fait, ordi-groz communique avec une borne Wifi qui est un pont ayant une interface radio Wifi et une interface Ethernet filaire pour la raccorder à hsp_lg.

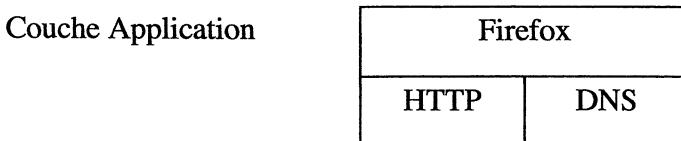
Le routage assuré par hsp_lg est filtrant : au départ, tant qu'un ordinateur n'a pas payé, hsp_lg ne fait sortir aucune communication sur l'Internet, et redirige tout vers ses propres serveurs, en fonction du port. Hsp_lg n'écoute en fait que sur 2 ports : 80 (HTTP) et 53 (DNS).

Lorsque ordi-groz lance son serveur Web Firefox dont la page d'accueil est voila.fr (193.252.148.80), c'est donc le serveur httpd de hsp_lg qui, jouant le rôle d'un 'proxy' récupère cette demande. Il répond en fournissant une réponse encapsulée dans un paquet IP paraissant venir de voila.fr (l'adresse IP source est bien 193.252.148.80) mais contenant un retour HTML :

La machine hotspots.Marron.fr est le portail 'Wifi payant' de Marron, d'adresse IP 80.10.46.232. La connexion TCP depuis le hotspot vers la machine hotspots.Marron.fr n'est pas filtrée, donc ordi-groz peut se connecter à cette page. Là il peut remplir un formulaire avec les éléments lui permettant de payer, et il achète un crédit de connexion de 1h. Lorsqu'il valide sur le formulaire son crédit, hotspots.Marron.fr informe (par un protocole applicatif de type RADIUS dont on ne détaillera pas les PDU) hsp_lg de ne plus filtrer les communications de ordi-groz, et renvoie vers ordi-groz la page de voila.fr qu'il avait initialement demandée.

Dans cette partie, vous devez représenter l'architecture statique en couches de toutes les machines mentionnées dans cet exemple (quelles que soient les couches qu'elles implémentent). On représentera les différentes couches, et on détaillera la couche application. Dans la couche application, lorsqu'une application utilise plusieurs protocoles, on représentera dans des rectangles superposés : le nom de l'application (par

ex Firefox, Webmail etc) et en dessous les protocoles qu'elle utilise, tels que http, DNS... Par exemple :



Précisez bien les adresses IP, les ports, et par des liens traversant les couches les connexions établies au cours de ce scénario.

Question 5. *Faites un diagramme d'architecture sur lequel figurent toutes les machines mentionnées ci-dessus, avec les couches, les applications et les protocoles.*

Déroulement des protocoles applicatifs pour l'accès (6 points)

Pour les questions demandant à illustrer le déroulement des protocoles applicatifs, on utilisera des diagrammes temporels (ou 'diagrammes de séquence') du type de ceux utilisés en cours pour illustrer le protocole du bit alternant ou la connexion TCP. On fera figurer un axe temporel (flèches verticales dans le cours) par machine impliquée. Sur les flèches en biais figurant les messages, on portera le type du PDU et une information minimale synthétique sur les paramètres essentiels par rapport au déroulement expliqué (par exemple pour le DNS : A ? suivi du nom d'une machine, A ? faisant référence à une 'query' de type 'Address').

Question 6. *Représentez sur un diagramme temporel l'enchaînement des PDUs des protocoles de la couche application (et uniquement eux) entre le moment où Roland Groz lance son navigateur et le moment où il voit apparaître la page d'accueil de Marron l'invitant à acheter du temps de connexion.*

Question 7. *Sur un diagramme temporel qui s'enchaîne avec le précédent, représentez les PDUs de couche application entre le moment où Roland Groz a confirmé son achat (démarrant son heure de crédit) et l'apparition de la page voila.fr sur son navigateur.*

Voila.fr était la page d'accueil, mais Roland Groz choisit maintenant dans ses signets (favoris, 'bookmarks') d'accéder à google.fr, d'adresse IP 66.249.93.104.

Question 8. *Représentez les PDUs de couche application jusqu'à l'affichage de la page de Google.*

Question de réflexion (2 points)

Question 9. *Pourquoi laisse-t-on dans un hotspot un accès au service DNS ? Il y a plusieurs éléments de réponse.*

Questions sur l'écoute en amphi (2 points + 1 point bonus)

Question de cours : signalisation téléphonique (1 point)

Question 10. Définir ce qu'on entend par "signalisation" en téléphonie et donner un exemple dans le cas du POTS et dans le cas du RNIS.

Ecoute mutuelle (1 point)

En début de cours, des étudiants ont eu à présenter des ingénieurs célèbres qui ont contribué aux techniques qui nous intéressent. Votre professeur a ainsi pu découvrir deux ingénieurs qui ont inspiré vos camarades.

Question 11. Pour les étudiants de l'amphi 1 (groupes 1,2,3) présentez Bertrand Serlet et pour ceux de l'amphi 2 (groupes 4,5,6) présentez Alexeï Pajitnov.
Seraient-ils des modèles pour vous-même ?

Question de réflexion : signalisation (1 point bonus)

Question 12. Alors qu'en téléphonie la signalisation joue un rôle essentiel, on n'en parle pas du tout dans les réseaux par paquets du type TCP/IP. Pourquoi ?
Qu'est-ce qui joue le rôle de la signalisation dans ces réseaux ?

EXAMEN DE “LOGIQUE POUR L’INFORMATIQUE”

Ensimag 1ère Année - mars 2009

DURÉE: 1 heure et demie.

SEULS DOCUMENTS AUTORISÉS: Poly “Une approche simple de la logique pour l’Informatique et la formalisation de l’inférence” et notes personnelles.

- S.V.P. avant de commencer lisez les 7 points ci-dessous.

- Cet examen comporte 5 questions. Toutes les questions sont indépendantes.
- L’ordre des questions n’a, en principe, aucun rapport avec leur difficulté.
- S.V.P. rédigez vos réponses de façon similaire à celle des corrigés du poly.
- **S.V.P. séparez** (par une ligne horizontale de la largeur de la feuille) et **numérotez clairement** les réponses aux différentes questions.
- **S.V.P.** respectez le nombre de lignes suggérées pour les réponses.
- Les réponses à l’aide des méthodes **autres** que celles demandées explicitement seront **ignorées**.
- On tiendra compte dans la correction de la clarté, de la simplicité et de la concision des réponses ainsi que de la qualité de la présentation.

Question 1

On se situe dans la LP.

On dit qu’une fbf de la LP est **positive**ssi elle contient **exclusivement** des symboles propositionnels et les connectifs \wedge et \vee .

Soit une fonction de vérité n-aire $f : \{ V, F \}^n \rightarrow \{ V, F \}$
telle que :

$f(F, F, \dots, F) = F$ % i.e. si tous ses arguments sont **F** alors sa valeur est **F**

$f(V, V, \dots, V) = V$ % i.e. si tous ses arguments sont **V** alors sa valeur est **V**
 f est la *table de vérité d’une fbf positive de la LP*.

- a) l’assertion ci-dessus est vraie.
- b) l’assertion ci-dessus est fausse.

Justifier : maxi 3 lignes.

Question 2

L’ensemble de formules S ci-dessous est-il satisfaisable ou insatisfaisable ? Trouver la réponse

en utilisant la méthode des tableaux sémantiques

$$S = \{ A \Leftrightarrow B \wedge \neg C, B \Leftrightarrow \neg A \vee C, C \Leftrightarrow A \wedge B \}$$

Question 3

L'ensemble de formules S ci-dessous est-il satisfaisable ou insatisfaisable ? Trouver la réponse

en utilisant la méthode de résolution

$$S = \{A \Leftrightarrow B \wedge \neg C, B \Leftrightarrow \neg A \vee C, C \Leftrightarrow A \wedge B\}$$

Question 4

G. Frege avait proposé un système formel $\mathcal{S}_F = < \mathcal{L}, \mathcal{R}, \mathcal{A} >$ avec:

\mathcal{L} : le langage L de S_1 poly, page 68

\mathcal{A} (i.e. les schémas d'axiomes):

(A_1): $A \Rightarrow (B \Rightarrow A)$

(A_2): $(C \Rightarrow (B \Rightarrow A)) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow A))$

(A_3): $((C \Rightarrow (B \Rightarrow A)) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow A))) \Rightarrow ((B \Rightarrow A) \Rightarrow ((C \Rightarrow (B \Rightarrow A)) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow A))))$

\mathcal{R} : {MP} modus ponens, poly page 68

a) Un logicien moderne affirme que (A_3) n'est pas indépendant. A-t-il raison ? A-t-il tort ? **Justifier.**

Question 5

Dans un calcul qu'on appelle *equivalential calculus* en anglais, les fbf du langage correspondant sont de la forme $e(X, Y)$ où e est une constante (pour ‘équivalent’) et X, Y des (méta-) variables dénotant des fbf du langage (comme A, B, C dans S_1 , poly, pages 68 et 69).

La seule règle d’inférence est :

$$CD : \frac{e(A, B)}{\sigma B} C \text{ avec } \sigma : upg(A, C)$$

Question :

Étant donné les deux fbf :

$$e(X, e(X, e(Y, Y))) \text{ et}$$

$$e(Z, Z)$$

1. Peut-on appliquer CD ?
 2. Combien de possibilités pour ce faire voyez-vous ?
 3. Si l'on peut appliquer CD , quel est(sont) la(les) conséquence(s) directe(s) ?
-

Fin-Examen

EXAMEN DE “LOGIQUE POUR L’INFORMATIQUE”

Ensimag 1ère Année - juin 2009

DURÉE: 1 heure et demie.

SEULS DOCUMENTS AUTORISÉS: Poly “Une approche simple de la logique pour l’Informatique et la formalisation de l’inférence” et notes personnelles.

- S.V.P. avant de commencer lisez les 7 points ci-dessous.

- Cet examen comporte 5 questions. Toutes les questions sont indépendantes.
- L’ordre des questions n’a, en principe, aucun rapport avec leur difficulté.
- S.V.P. rédigez vos réponses de façon similaire à celle des corrigés du poly.
- **S.V.P. séparez** (par une ligne horizontale de la largeur de la feuille) et **numérotez clairement** les réponses aux différentes questions.
- **S.V.P.** respectez le nombre de lignes suggérées pour les réponses.
- Les réponses à l’aide des méthodes **autres** que celles demandées explicitement seront **ignorées**.
- On tiendra compte dans la correction de la clarté, de la simplicité et de la concision des réponses ainsi que de la qualité de la présentation.

Question 1

Dans l’arithmétique de Presburger (voir poly, Exemple 39, page 122), c’est à dire :

$$\mathbb{N}_A = \langle \mathbb{N}, 0, \text{succ}, +, = \rangle$$

On veut définir ‘<’ i.e. la relation inférieur entre naturels.

On propose :

- $x < y : \text{def } \forall x \forall y \exists z (x + z = y)$
- $x < y : \text{def } \exists z (x + z = y)$

Laquelle vous semble correcte ?

- ?
- ?
- les deux ?
- aucune des deux ?

Justifier. (Maxi 3 lignes).

Question 2

Transformer en forme clausale :

$$\neg [\exists u \forall z (P(u) \wedge (P(z) \Rightarrow Q(z, u)))]$$

Question 3

Pouvez-vous donner une fbf **4-valide** de la L1O ?

Aide : Inspirez-vous de l'exemple 46, page 134 poly

Question 4

Soit le raisonnement suivant, exprimé en L1O et, comme d'habitude a , f et M dénotent respectivement une constante, un symbole de fonction et un symbole de prédicat :

$$\begin{aligned} \forall x(M(x) \Leftrightarrow \neg M(f(x))) \\ M(a) \vee M(f(f(f(a)))) \Rightarrow M(f(a)) \end{aligned}$$

$$\exists y \neg M(y)$$

Dire s'il est correct, incorrect ou si l'on ne peut rien dire,
en utilisant la règle de résolution :

Indiquer clairement les pas de résolution, comme dans les corrigés du poly.

Question 5

Soit le raisonnement suivant, exprimé en L1O et, comme d'habitude a , f et M dénotent respectivement une constante, un symbole de fonction et un symbole de prédicat :

$$\begin{aligned} \forall x(M(x) \Leftrightarrow \neg M(f(x))) \\ M(a) \vee M(f(f(f(a)))) \Rightarrow M(f(a)) \end{aligned}$$

$$\exists y \neg M(y)$$

Dire s'il est correct, incorrect ou si l'on ne peut rien dire,

en utilisant la méthode des tableaux sémantiques avec unification :

Indiquer clairement la construction de tableau, comme dans les corrigés du poly.

Fin-Examen

EXAMEN DE “LOGIQUE POUR L’INFORMATIQUE”

Ensimag 1ère Année - mai 2009

DURÉE: 1 heure et demie.

SEULS DOCUMENTS AUTORISÉS: Poly “Une approche simple de la logique pour l’Informatique et la formalisation de l’inférence” et notes personnelles.

- S.V.P. avant de commencer lisez les 7 points ci-dessous.

- Cet examen comporte 6 questions. Toutes les questions sont indépendantes.
- L’ordre des questions n’a, en principe, aucun rapport avec leur difficulté.
- S.V.P. rédigez vos réponses de façon similaire à celle des corrigés du poly.
- **S.V.P. séparez** (par une ligne horizontale de la largeur de la feuille) et **numérotez clairement** les réponses aux différentes questions.
- **S.V.P.** respectez le nombre de lignes suggérées pour les réponses.
- Les réponses à l’aide des méthodes **autres** que celles demandées explicitement seront **ignorées**.
- On tiendra compte dans la correction de la clarté, de la simplicité et de la concision des réponses ainsi que de la qualité de la présentation.

Question 1

Quelqu’un prétend que l’on peut définir l’arithmétique de Presburger (voir poly, Exemple 39, page 122), c’est à dire :

$$\mathbb{N}_A = \langle \mathbb{N}, 0, succ, +, =, < \rangle$$

par la structure suivante :

$$\mathbb{N}_A = \langle \mathbb{N}, 0, succ, +, = \rangle$$

A-t-il raison ou tort ? Justifier. (Maxi 3 lignes).

Question 2

Soit le raisonnement suivant :

$$\begin{aligned} \forall x[H(x) \Rightarrow A(x)] \\ \exists y \forall v[H(y) \wedge (A(v) \Rightarrow L(v, y))] \end{aligned}$$

$$\exists u \forall z[H(u) \wedge (H(z) \Rightarrow L(z, u))]$$

Dire si ce raisonnement est correct ou incorrect

en utilisant la méthode des tableaux sémantiques (avec unification)

Question 3

Soit le raisonnement suivant :

$$\begin{aligned} \forall x[H(x) \Rightarrow A(x)] \\ \exists y \forall v[H(y) \wedge (A(v) \Rightarrow L(v, y))] \end{aligned}$$

$$\exists u \forall z[H(u) \wedge (H(z) \Rightarrow L(z, u))]$$

Dire si ce raisonnement est correct ou incorrect

en utilisant la méthode de résolution

Question 4

Considérer l'ensemble $\mathcal{C} = \{S \mid S \text{ ensemble de clauses unitaires}\}$

et l'assertion suivante :

“La méthode de résolution peut décider si S sat ou insat pour tout $S \in \mathcal{C}$ ”

Est-elle vraie ou fausse ?

Justifier. Maxi 3 lignes.

Question 5

On souhaite définir la notion de **densité**, i.e. dans un ensemble totalement ordonné (e.g. \mathbb{Q}), étant donné 2 éléments (l'un plus grand que l'autre) il existe toujours un élément entre les deux.

Des 2 formalisations ci-dessous :

- a) densité: $\forall x \forall y (x < y \Rightarrow \exists z (x < z < y))$
- b) densité: $\forall x \forall y \exists z (x < y \Rightarrow (x < z < y))$

Laquelle vous semble correcte

- a) ?
- b) ?
- c) les deux ?
- d) aucune des deux ?

Justifier. Maxi 3 lignes.

Question 6

On dispose d'un graphe dont les noeuds représentent des villes et les arcs des vols directs existant entre deux villes (étiquettés, disons, $\text{Vol}(\text{ville-i}, \text{ville-j})$).

On dira qu'une connexion aérienne entre deux villes est *acceptable* si l'on peut aller de l'une à l'autre avec au plus 2 arrêts (i.e. 3 vols directs).

On veut pouvoir poser des questions du genre :

Vol-acceptable(ville-a, ville-b) ;

Pouvez-vous donner la clause de Horn qu'il faut ajouter au programme logique pour qu'il puisse donner les réponses ? **Maximum 5 lignes.**

Fin-Examen

Logiciel de Base : examen de première session

ENSIMAG 1A

Année scolaire 2008–2009

Consignes générales :

- Durée : 2h. Tous documents et calculatrices autorisés.
- Le barème est donné à titre indicatif.
- Les exercices sont indépendants et peuvent être traités dans le désordre.
- **Merci d'indiquer votre numéro de groupe de TD et de rendre votre copie dans le tas correspondant à votre groupe.**

Consignes relatives à l'écriture de code C et assembleur Pentium :

- Pour chaque question, une partie des points sera affectée à la clarté du code et au respect des consignes ci-dessous.
- Pour les questions portant sur la traduction d'une fonction C en assembleur, on demande d'indiquer en commentaire chaque ligne du programme C original avant d'écrire les instructions assembleur correspondantes.
- Pour améliorer la lisibilité du code assembleur, on utilisera systématiquement des constantes (directive `.equ`) pour les déplacements relatifs à `%ebp` (*i.e.* paramètres des fonctions et variables locales). Par exemple, si une variable locale s'appelle `var` en langage C, on y fera référence avec `var(%ebp)`.
- Sauf indication contraire dans l'énoncé, on demande de traduire le code C en assembleur de façon systématique, sans chercher à faire la moindre optimisation : en particulier, **on stockera les variables locales dans la pile** (pas dans des registres), comme le fait le compilateur C par défaut.
- On respectera les conventions de gestions des registres Intel vues en cours, c'est à dire :
 - `%eax`, `%ecx` et `%edx` sont des registres *scratch* ;
 - `%ebx`, `%esi` et `%edi` ne sont pas des registres *scratch*.

Ex. 1 : Parcours de tableaux (12 pts)

Le Crible d'Ératosthène¹ est un algorithme itératif permettant de générer les nombres premiers inférieurs ou égaux à un entier N donné.

On travaille ici sur une version de cet algorithme utilisant des tableaux. Plus précisément, l'algorithme consiste à :

¹Ératosthène était un astronome, géographe, philosophe et mathématicien grec du III^e siècle avant J.-C.

- initialiser un tableau de booléens allant de 0 à N (inclus) avec la valeur *Faux* pour les cases 0 et 1 (qui ne sont pas des nombres premiers) et *Vrai* pour les autres ;
- parcourir ce tableau pour marquer avec la valeur *Faux* toutes les cases dont l'indice est un multiple d'un entier inférieur : on démontre qu'on peut arrêter le parcours dès qu'on a supprimé tous les multiples des entiers inférieurs ou égaux à \sqrt{N} ;
- afficher les indices des cases contenant la valeur *Vrai*, car les indices en question sont tous des nombres premiers.

Le programme principal de cet algorithme s'écrit donc en C (où N est un naturel supérieur ou égal à 2) :

```
int main()
{
    unsigned int tab[N + 1]; /* unsigned int est équivalent à unsigned */
    remplir(tab, N);
    supprimer_multiples(tab, N);
    afficher(tab, N);
    return 0;
}
```

Question 1 Ecrire en C la fonction `void remplir(unsigned int tab[], unsigned int sup)` qui initialise le tableau `tab` comme expliqué plus haut (on rappelle qu'en C il n'existe pas de type booléen : on utilisera donc 0 pour la valeur *Faux* et 1 pour la valeur *Vrai*).

Question 2 Traduire cette fonction `remplir` en assembleur. On rappelle qu'il est demandé de précéder chaque séquence d'instructions assembleur par la ligne de C correspondante, et d'utiliser des constantes pour les déplacements relatifs à `%ebp`.

Question 3 Traduire en C le code de la fonction `supprimer_multiples` donnée ci-dessous en assembleur. On conservera les mêmes noms de paramètres et de variables locales lors de la traduction. On rappelle quelques instructions utiles pour cette question :

- `enter $8, $0` : cette instruction est exactement équivalente à la séquence d'instructions :

```
pushl %ebp
movl %esp, %ebp
subl $8, %esp
```

- `mull x` : multiplie le contenu de la variable 32 bits `x` par le contenu du registre `%eax`, et stocke le résultat (sur 64 bits) dans `%edx:%eax` ;
- `shll %eax` : décale le contenu du registre `%eax` d'un bit vers la gauche (ce qui revient en pratique à le multiplier par 2).

```
/* void supprimer_multiples(unsigned int tab[], unsigned int sup) */
.equ tab, 8
.equ sup, 12
supprimer_multiples:
/* unsigned int i, j */
```

```

.equ i, -4
.equ j, -8
enter $8, $0
movl $2, i(%ebp)
sm_bcl1:
    movl i(%ebp), %eax
    mull i(%ebp)
    cmpl sup(%ebp), %eax
    ja sm_bcl1_fin
    movl i(%ebp), %eax
    shll %eax
    movl %eax, j(%ebp)
sm_bcl2:
    movl j(%ebp), %eax
    cmpl sup(%ebp), %eax
    ja sm_bcl2_fin
    movl tab(%ebp), %eax
    movl j(%ebp), %edx
    movl $0, (%eax, %edx, 4)
    movl i(%ebp), %eax
    addl %eax, j(%ebp)
    jmp sm_bcl2
sm_bcl2_fin:
    incl i(%ebp)
    jmp sm_bcl1
sm_bcl1_fin:
    leave
    ret

```

Question 4 Ecrire en C la fonction `void afficher(unsigned int tab[], unsigned int sup)` qui affiche à l'écran les nombres premiers en fonction du contenu du tableau `tab`, comme expliqué plus haut.

Ex. 2 : Optimisation de code (8 pts)

On travaille dans cet exercice sur différentes implantations d'une fonction permettant de copier le contenu d'une zone mémoire dans une autre. Cette fonction existe dans la bibliothèque C standard :

```

/*
 * Recopie les n premiers octets de la zone memoire pointee par src
 * dans la zone memoire pointee par dst et renvoie l'adresse dst.
 */
char *memcpy(char *dst, char *src, unsigned int n)

```

Une implantation simpliste de cette fonction en C pourrait être :

```

char *copie_memoire(char *dst, char *src, unsigned int n)
{
    unsigned int i;
    for (i = 0; i < n; i++)
        dst[i] = src[i];
    return dst;
}

```

Question 1 Traduire cette version simpliste en assembleur. On traduira de façon systématique sans chercher à faire la moindre optimisation. Notamment, les variables locales seront stockées dans la pile et pas dans des registres.

Question 2 Ecrire une nouvelle version assembleur de cette même fonction, mais en stockant au début du programme les variables locales et les paramètres dans des registres, pour supprimer dans la suite du code autant d'accès mémoire que possible. On stockera `src` dans `%esi`, `dst` dans `%edi`, `n` dans `%edx` et `i` dans `%ecx`.

En pratique, l'implantation de `memcpy` fournie par la bibliothèque C standard est plus rapide que les deux versions implantées dans les questions précédentes. On peut effectivement optimiser le code en utilisant des instructions dédiées du processeur Pentium :

- `shrl $X, %reg` : décale de `X` bits vers la droite le registre `reg` ;
- `movsd` : recopie le mot de 32 bits pointé par `%esi` dans la case mémoire pointée par `%edi` et ajoute ou soustrait 4 à `%esi` et `%edi` (le choix entre l'addition ou la soustraction se fait en fonction du bit DF du registre `%eflags`) ;
- `cld` : met à zéro le bit DF du registre `%eflags`, ce qui a pour effet en pratique de sélectionner l'addition de 4 à `%esi` et `%edi` pour l'instruction `movsd` ;
- `rep` : préfixe ajouté avant certaines instructions (*e.g.* `rep movsd`) et qui a pour effet de répéter l'exécution de l'instruction préfixée tant que `%ecx` est différent de 0, et qui soustrait automatiquement 1 à `%ecx` après chaque exécution de l'instruction.

Question 3 En utilisant les instructions présentées ci-dessus, donner une version assembleur optimisée de la fonction de copie mémoire. On supposera pour simplifier que la taille de la zone mémoire à recopier est un multiple de 4.

Pour les curieux, un test simpliste effectué sur `tellesun` (recopie d'une zone mémoire de 1468006400 octets) donne les résultats suivants : version naïve en 5,98 sec., version optimisée en 1,93 sec., version avec les instructions dédiées et `memcpy` de la bibliothèque C toutes deux en 1,80 sec. En pratique, on copie rarement de très gros blocs de mémoire, mais plus souvent de petits blocs de façon répétitives. L'implantation de `memcpy` de la bibliothèque C est alors très performante grâce à la prise en compte des caractéristiques du pipeline et de la mémoire cache de la machine.

Examen du 20 Mai 2009**Durée : 3h.**

Les seuls documents autorisés sont les notes du cours et des travaux dirigés de méthodes numériques.

Exercice 1 On étudie dans cet exercice une méthode de résolution d'un système linéaire $Ax = b$, avec $A \in M_n(\mathbb{R})$ inversible et $b, x \in \mathbb{R}^n$. Cette méthode consiste à se ramener au système $A^t Ax = A^t b$, puis à le résoudre par la méthode de Cholesky.

- 1- Montrer que la matrice $A^t A$ est symétrique définie positive.
- 2- Lorsque $n \rightarrow +\infty$, donner un équivalent du nombre d'opérations arithmétiques élémentaires nécessaires à la résolution du système $Ax = b$ avec cette méthode.
- 3- Comparer les coûts de l'algorithme précédent et de la résolution directe de $Ax = b$ par la méthode de Gauss lorsque n est grand.

Exercice 2 Déterminer si les méthodes de Jacobi et Gauss-Seidel convergent pour le système linéaire $Ax = b$ avec

$$A = \begin{pmatrix} 1 & 3/4 & 3/4 \\ 3/4 & 1 & 3/4 \\ 3/4 & 3/4 & 1 \end{pmatrix}$$

et $b = (1, \frac{3}{4}, 1)^t$.

La résolution du système linéaire n'est pas demandée.

Exercice 3 Dans ce problème on étudie un système modélisant un circuit électrique qui comporte $n+1$ résistances identiques R . On note x_k le potentiel électrique au noeud reliant les k ième et $(k+1)$ ième résistances, et j_k l'intensité d'un courant d'entrée en ce point. Le circuit est construit de manière à avoir

$$2x_k - x_{k+1} - x_{k-1} = R j_k, \quad k = 1, \dots, n, \quad (1)$$

$$x_0 = x_{n+1} = 0, \quad (2)$$

avec $R > 0$. On considère pour l'instant j_1, \dots, j_n comme des paramètres et on souhaite calculer x_1, \dots, x_n . En introduisant $x = (x_1, \dots, x_n)^t \in \mathbb{R}^n$, $b = R(j_1, \dots, j_n)^t \in \mathbb{R}^n$ et la matrice $A \in M_n(\mathbb{R})$ suivante

$$A = \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & \vdots \\ 0 & -1 & \ddots & \ddots & 0 \\ \vdots & 0 & \ddots & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{pmatrix}, \quad (3)$$

le problème (1)-(2) s'écrit sous la forme $Ax = b$. On rappelle l'expression des valeurs propres de A

$$\lambda_m = 4 \sin^2 \left(\frac{m\pi}{2(n+1)} \right), \quad m = 1, \dots, n.$$

Partie I

Dans cette partie on étudie le conditionnement κ_∞ de A pour la norme $\|\cdot\|_\infty$.

1. Calculer $\|A\|_\infty$.
2. Sans calculer explicitement A^{-1} , montrer que

$$\|A^{-1}\|_\infty \leq c_n \frac{(n+1)^2}{8}, \quad c_n = \begin{cases} 1 & \text{si } n \text{ est impair,} \\ 1 - \frac{1}{(n+1)^2} & \text{si } n \text{ est pair.} \end{cases}$$

3. En utilisant l'expression des valeurs propres de A , trouver une constante $c > 0$ telle que $\|A^{-1}\|_\infty \geq c(n+1)^2$.
4. En déduire $\alpha, \beta > 0$ tels que $\alpha(n+1)^2 \leq \kappa_\infty \leq \beta(n+1)^2$. La matrice A est-elle bien conditionnée pour la norme $\|\cdot\|_\infty$ lorsque $n \rightarrow +\infty$?
5. Montrer que

$$\|A^{-1}\|_\infty = c_n \frac{(n+1)^2}{8}.$$

Indication : on pourra introduire la solution du système $Ax = b$ avec $b_i = 1$ pour tout $i = 1, \dots, n$.

6. En déduire la valeur exacte du conditionnement de A pour la norme $\|\cdot\|_\infty$.

- 7.** On fixe dans cette question $n = 19$. On considère des données d'entrée $b + \delta b$ sur lesquelles on commet une erreur de mesure $\delta b \in \mathbb{R}^n$ pour laquelle $\|\delta b\|_\infty / \|b\|_\infty \leq 10^{-4}$. On considère la solution $x + \delta x$ du système $A(x + \delta x) = b + \delta b$. Donner une majoration de l'erreur relative $\|\delta x\|_\infty / \|x\|_\infty$.

Partie II

Dans cette partie on étudie d'un point de vue général la résolution numérique de systèmes linéaires dont la matrice est tridiagonale comme la matrice (3). On considère une matrice $M \in M_n(\mathbb{R})$ inversible et tridiagonale

$$M = \begin{pmatrix} a_1 & c_2 & 0 & \cdots & 0 \\ b_2 & a_2 & c_3 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & b_{n-1} & a_{n-1} & c_n \\ 0 & \cdots & 0 & b_n & a_n \end{pmatrix}$$

dont tous les coefficients sont supposés non nuls. On suppose que M admet une factorisation $M = LU$, avec $L \in M_n(\mathbb{R})$ triangulaire inférieure de diagonale unité, et $U \in M_n(\mathbb{R})$ triangulaire supérieure. Dans ce cas on peut montrer par récurrence que les matrices L et U sont bidiagonales et s'écrivent

$$L = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ \gamma_2 & 1 & 0 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & \gamma_{n-1} & 1 & 0 \\ 0 & \cdots & 0 & \gamma_n & 1 \end{pmatrix}, \quad U = \begin{pmatrix} \alpha_1 & c_2 & 0 & \cdots & 0 \\ 0 & \alpha_2 & c_3 & 0 & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & 0 & 0 & \alpha_{n-1} & c_n \\ 0 & \cdots & 0 & 0 & \alpha_n \end{pmatrix} \quad (4)$$

(on ne demande pas de démontrer ce résultat).

- 8.** Donner une relation de récurrence qui détermine les coefficients γ_k et α_k des matrices L et U .

- 9.** Lorsque $n \rightarrow +\infty$, donner un équivalent du nombre d'opérations arithmétiques élémentaires nécessaires au calcul de L et U ainsi qu'à la résolution d'un système $Mx = b$ ($x, b \in \mathbb{R}^n$) utilisant cette factorisation LU .

- 10.** Lorsque M est symétrique définie positive, montrer qu'il existe une matrice diagonale $D \in M_n(\mathbb{R})$ et une matrice $L \in M_n(\mathbb{R})$ de la forme donnée

dans (4) telles que $M = L D L^t$. Donner une relation de récurrence qui détermine les coefficients des matrices L et D .

11. Pour la matrice A donnée par (3), calculer explicitement les matrices L et D intervenant dans la factorisation $A = L D L^t$.

Partie III

On revient à l'étude du problème (1)-(2). On suppose maintenant que

$$j_k = \frac{1}{R_k} (v_k - x_k), \quad k = 1, \dots, n, \quad (5)$$

où v_1, \dots, v_n sont des paramètres correspondant à des tensions d'entrée et $R_k > 0$ des résistances. On note par la suite $v = (v_1, \dots, v_n)^t$.

12. Montrer que le problème (1)-(2)-(5) conduit à résoudre un système linéaire $M x = D v$ pour $x = (x_1, \dots, x_n)^t \in \mathbb{R}^n$, avec $D \in M_n(\mathbb{R})$ diagonale et $M \in M_n(\mathbb{R})$ (on explicitera les matrices D et M).

13. Montrer que la matrice M est symétrique définie positive.

14. On souhaite résoudre (1)-(2)-(5) pour un grand nombre de valeurs du vecteur v contenant les tensions d'entrée. Quelle méthode numérique choisissez-vous pour résoudre ce problème ? On justifiera soigneusement le choix de la méthode en question.

15. On remplace maintenant la relation (5) par

$$j_k = \rho_k(v_k - x_k), \quad k = 1, \dots, n, \quad (6)$$

où les fonctions $\rho_k : \mathbb{R} \rightarrow \mathbb{R}$ sont de classe C^2 et donnent les caractéristiques courant-tension de résistances non linéaires. Donner une méthode numérique pour résoudre (1)-(2)-(6).

16. On suppose $v_k = v$ et $\rho_k = \rho$ indépendants de k , avec $\rho(x) = \operatorname{sh} x$. Le problème (1)-(2)-(6) s'écrit alors

$$x_{k+1} - 2x_k + x_{k-1} = R \operatorname{sh}(x_k - v), \quad k = 1, \dots, n, \quad (7)$$

avec $x_0 = x_{n+1} = 0$. Montrer que ce problème admet une solution et que celle-ci est unique.

Indication : on pourra se ramener à un problème de minimisation.

Examen : Ondes et transmissions

Ensimag
1^{ère} année

Partie 1 : propagation

On considère un milieu isolant parfait, de constante diélectrique ϵ et de perméabilité magnétique μ , sans sources de charges ou de courants, dans lequel se propage un champ électrique de pulsation ω , dont le vecteur complexe $\vec{E}(x,y,z)$ ne dépend que de z .

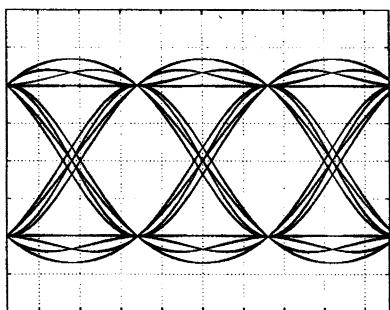
- 1) Quelle est l'équation de propagation de \vec{E} ?
- 2) Donner la valeur de la constante γ pour que \vec{E} soit de la forme : $\vec{E} = \vec{E}_0 \exp(-\gamma z)$, avec \vec{E}_0 un vecteur complexe constant.
- 3) Comment s'exprime le champ électrique réel en fonction de \vec{E} ?
- 4) Donner les composantes de \vec{E}_0 sachant que le champ réel mesuré en $z=0$ aux instants $t_0=0$ et $t_1=0,25 \times (2\pi/\omega)$ vaut :

$$\vec{E}(z=0, t=t_0) = \begin{cases} 0 \\ 1 \end{cases} \text{ et } \vec{E}(z=0, t=t_1) = \begin{cases} 0 \\ 1 \\ 0 \end{cases}$$

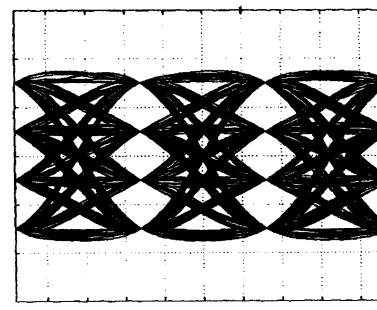
Partie 2 : systèmes de transmission radio

- 5) Avec un signal modulé avec une modulation QAM16, quelle forme de diagramme de l'œil trouve-t-on sur les voies en phase et en quadrature, a ou b ?

cas a :



cas b :



La figure suivante donne le schéma d'un émetteur de faisceaux hertziens.

- 6) D'après le schéma, dire quelle est la modulation utilisée dans cet émetteur en expliquant la réponse.
- 7) Expliquer le rôle du deuxième mélangeur sur lequel est appliqué le signal d'oscillateur local OL_2 .

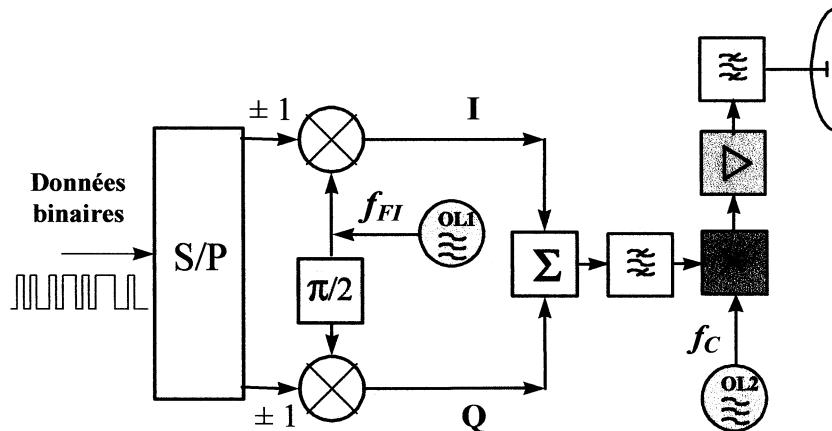


Figure : Emetteur de faisceaux hertziens

En ADSL, pour la transmission entre le modem et le DSLAM (multiplexeur de lignes ADSL), une modulation DMT (Discrete MultiTone) est utilisée. Cette dernière consiste à moduler un ensemble de porteuses régulièrement espacées de 4,3125 kHz avec des modulations QAM. Le codage est de type 'orthogonal' ce qui signifie notamment que la rapidité de modulation sur chaque canal correspond exactement à l'espacement des différentes porteuses.

Supposons que le canal n°38 utilise une modulation QAM64, suffisante pour assurer un taux d'erreurs binaires correct (de l'ordre de 10^{-7}).

- 8) Sachant que le codage est orthogonal, donner la valeur de la rapidité de modulation sur chaque canal. Que vaut alors le débit binaire sur le canal n°38 ?

Suite à des interférences sur la ligne, la puissance de bruit augmente de 6 dB sur ce canal.

- 9) Quel paramètre permet d'estimer la dégradation de taux d'erreurs binaires ? Donner sa définition et sa valeur sur le canal 38 suite aux interférences.
- 10) Expliquer pourquoi si l'on veut conserver la même qualité de transmission sur ce canal, il faut changer de modulation. Suffit-il de choisir une modulation QAM16 ? (Justifier toutes les étapes du raisonnement.) Si non, proposer une solution. De quel facteur aura finalement baissé le débit de données sur ce canal ?

- 11) Quelle est la conséquence néfaste du phénomène de trajets multiples dans un système de transmission radio ? Donner deux exemples de systèmes de transmission où ces effets interviennent de manière importante.

- 12) Citer des méthodes que l'on peut utiliser pour diminuer l'impact de ces trajets multiples.

Partie 3 : systèmes de transmission guidée

- 13) Quels sont les trois principaux types d'interaction entre les photons et les électrons dans un semiconducteur ? Pour chaque type d'interaction, citer un

composant optoélectronique dont le principe de fonctionnement repose directement sur elle.

La norme CEI 153 spécifie les caractéristiques des guides d'ondes métalliques rectangulaires à utiliser. Par exemple, pour le guide R100, la norme donne :

Bandes de fréquences mode fondamental		Dimensions du guide		Affaiblissement	
f_{\min} (GHz)	f_{\max} (GHz)	a (mm)	b (mm)	à f (GHz)	théorique (dB/m)
8,20	12,5	22,860	10,160	9,84	0,110

- 14) Expliquer pourquoi une fréquence minimale et une fréquence maximale sont spécifiées pour chaque type de guide. En particulier, que se passe-t-il si on utilise une fréquence inférieure à la fréquence minimale ou une fréquence supérieure à la fréquence maximale ?**

On considère une ligne coaxiale idéale constituée d'un isolant de constante diélectrique relative $\epsilon_r=2,32$ (la longueur d'onde vaut alors sur cette ligne $\lambda = \frac{c}{\sqrt{\epsilon_r} f}$) et d'impédance caractéristique 50Ω . La charge terminale est constituée d'une résistance $R=100 \Omega$ en série avec une inductance $L=100 \text{ nH}$. A $f=500 \text{ MHz}$:

- 15) Déterminer l'impédance réduite dans le plan de charge et positionner la sur l'abaque de Smith fournie.**
- 16) Donner la valeur du module du coefficient de réflexion sur la ligne. La puissance fournie par le générateur est-elle transmise de manière efficace à la charge ?**
- 17) Trouver la distance la plus courte par rapport au plan de charge qui permette d'obtenir une impédance équivalente réelle.**
- 18) Quels avantages les fibres optiques présentent-elles par rapport aux supports de transmission à câbles conducteurs ?**

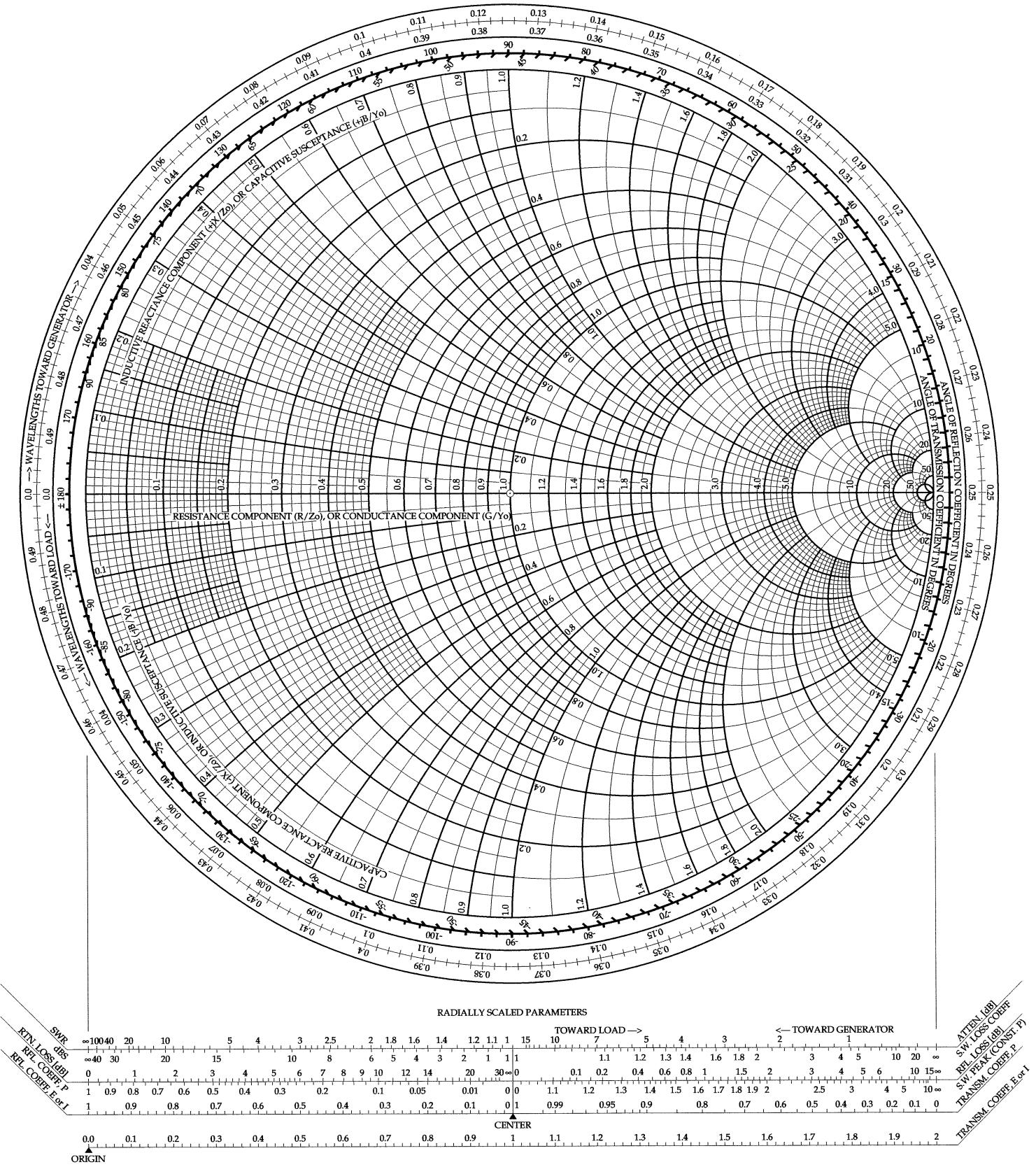
Partie 4 : protocoles des liens de transmission

Soit un échange de données effectué avec le protocole HDLC. Etudions le cas où les données devant être transmises par ce protocole sont : 101011111010010.

- 19) Ces données seront-elles transmises simplement en les insérant dans le champ 'données' de la trame HDLC, sans aucune modification ? Si non, pourquoi ? Quels seraient alors les bits effectivement transmis dans ce champ 'données' ? Expliquer le principe du mécanisme mis en jeu.**
- 20) Qu'appelle-t-on une 'technique d'accès' dans un système de télécommunication ? Quelles sont les techniques d'accès utilisées dans le GSM et dans l'UMTS ? Expliquer brièvement leur principe.**

The Complete Smith Chart

Black Magic Design



Probabilités Appliquées
Novembre 2008

Durée 2h00 - Documents, calculatrices, téléphones portables non autorisés. Veuillez inscrire votre numéro de groupe sur la copie.

1. (2 pts) Décrire une situation générique dans laquelle on peut observer une variable aléatoire de loi binomiale $B(n, p)$. Donner un argument simple pour justifier que la somme d'une variable aléatoire de loi $B(n_1, p)$ et d'une variable aléatoire de loi $B(n_2, p)$ indépendante de la première est une variable aléatoire de loi $B(n_1 + n_2, p)$.

2. (4 pts) Dans le championnat de basketball de l'Uhgduzstan, il y a 1/3 de tirs à un point, 1/2 de tirs à deux points et 1/6 de tirs à 3 points. Vlad Rabovitch est le meilleur joueur du pays. Sa fréquence de réussite est à 1 point de 1/2, à 2 points de 1/3, à 3 points de 1/4.
 - a) Quelle est la probabilité que Vlad réussisse un tir?
 - b) Vlad vient de rater un tir. Quelle est la probabilité qu'il ait tiré à 3 points?
 - c) Vlad vient de réussir un tir. Quelle est la probabilité qu'il ait tiré à 3 points?
 - d) Combien de tirs réussis doit on attendre en moyenne avant de voir Vlad marquer à 3 points?

3. (2 pts) La durée de connexion sur le serveur de l'ensimag, notée Z , obéit à la loi
$$\forall t \in \mathbb{R}_+, \quad P(Z \leq t) = 1 - \exp(-\lambda t), \quad \lambda > 0,$$
où le temps t est exprimé en heures. L'administrateur s'intéresse aux connexions de durée supérieure à 2 heures. On note Z_2 l'une de ces durées. Quelle est la loi de la variable aléatoire Z_2 ?

4. (3 pts) On joue à pile ou face jusqu'à obtenir pile avec une pièce équilibrée. Cela nécessite N lancers. On note la valeur N et on tire un nombre aléatoire entier, X , uniformément réparti dans l'ensemble $\{1, \dots, N\}$.
- Montrer que la probabilité de l'événement $(X = 1)$ est égale à $\log 2$.
 - Calculer la probabilité $P(X = k)$, $k \geq 1$.

Rappel: Pour tout $z \in [0, 1)$, $-\log(1 - z) = \sum_{n=1}^{\infty} \frac{z^n}{n}$.

5. (3 pts) On tire U un nombre au hasard uniformément dans l'intervalle $(0, 1)$, puis on lance une pièce équilibrée dont le résultat est indépendant de U . Si c'est pile, on définit $X = U$ sinon, $X = 2$. Soit $t \in [0, 2]$, calculer la probabilité de l'événement $(X \leq t)$. Représenter graphiquement la fonction qui à t associe $P(X \leq t)$.
6. (2 pts) Soit α, β deux nombres réels positifs. On considère une variable aléatoire, X , égale à α avec la probabilité $1/3$ et à β avec la probabilité $2/3$. En introduisant une variable aléatoire U prise au hasard uniformément dans l'intervalle $(0, 1)$, représenter X comme une variable étagée. Calculer son espérance.
7. (2 pts) Soit X et Y deux variables aléatoires indépendantes à valeurs dans $\{1, \dots, m\}$ et $\{1, \dots, n\}$ respectivement. En utilisant les événements $(X = i)$, $(Y = j)$, montrer que l'on peut écrire X , Y et XY comme des variables aléatoires étagées et démontrer que
- $$E[XY] = E[X] \times E[Y].$$
8. (2 pts) Soit X une variable de loi $B(1, p_1)$ et N une variable aléatoire de loi géométrique de paramètre p_2 , indépendante de X . Déterminer la loi de la variable $Z = 1 + XN$. Que vaut son espérance?

Probabilités Appliquées 2
Mai 2009

Durée 2h00 - Documents et calculatrices non autorisés.

I. Exercice

On considère un vecteur gaussien X en dimension 3 de moyenne nulle et de matrice de covariance

$$K = \begin{pmatrix} \sigma_1^2 & 0 & c_{13} \\ 0 & \sigma_2^2 & c_{23} \\ c_{13} & c_{23} & \sigma_3^2 \end{pmatrix}$$

où $\sigma_i > 0$. Les coordonnées de X sont notées X_1 , X_2 et X_3 . On définit le vecteur Y de la manière suivante

$$\begin{cases} Y_1 = X_1 \\ Y_2 = X_2 \\ Y_3 = X_3 - c_{13}X_1/\sigma_1^2 - c_{23}X_2/\sigma_2^2 \end{cases}$$

1. Montrer que le vecteur Y est gaussien et de moyenne nulle.
2. Calculer $\text{Cov}(Y_1, Y_3)$ et $\text{Cov}(Y_2, Y_3)$. Montrer que les coordonnées Y_1 , Y_2 et Y_3 sont indépendantes.
3. Montrer que

$$\text{Var}(X_3) = \text{Var}(Y_3) + c_{13}^2/\sigma_1^2 + c_{23}^2/\sigma_2^2$$

En déduire la loi de Y .

4. On dispose d'un générateur aléatoire `norm()` retournant des variables indépendantes de loi $\mathcal{N}(0, 1)$. Écrire un algorithme de simulation d'un vecteur gaussien de moyenne $m = (0, 1, 0)$ et de matrice de covariance K .

II. Problème

On considère une suite (U_n) de variables aléatoires indépendantes de loi uniforme sur $(0, 1)$ et la fonction

$$\forall u \in (0, 1), \quad \varphi(u) = \sqrt{(1-u)u^3}.$$

1. Pour tout $n \geq 1$, on pose

$$Y_n = \frac{1}{n} \sum_{i=1}^n \varphi(U_i).$$

Justifier le fait que la suite Y_n converge p.s. vers la limite \mathcal{I} définie ci-dessous

$$\mathcal{I} = \int_0^1 \varphi(u) du.$$

On admettra que $\mathcal{I} = \pi/16$.

2. Calculer la variance de la variable aléatoire Y_n .
3. Soit $\epsilon = 10^{-3}$. À l'aide du théorème de tendance vers la loi normale, donner une estimation du rang n à partir duquel on peut considérer que

$$P(|Y_n - \mathcal{I}| < \epsilon) \geq 0.95.$$

(Rappel : La fonction de répartition de la loi normale $F(t)$ est approximativement égale à 0.975 au point $t = 1.96$).

4. On considère la loi de densité f définie sur l'intervalle $(0, 1)$ de la manière suivante

$$\forall v \in (0, 1), \quad f(v) = 6v(1-v).$$

et une suite (V_n) de variables aléatoires indépendantes et de loi de densité f . Pour tout $n \geq 1$, on pose

$$Z_n = \frac{1}{n} \sum_{i=1}^n \frac{\varphi(V_i)}{f(V_i)}.$$

Montrer que la suite Z_n converge p.s. vers \mathcal{I} et comparer la variance de la variable Z_n à celle de Y_n .

5. Écrire un algorithme de simulation de la loi de densité f .
6. Proposer deux algorithmes de calcul de l'intégrale \mathcal{I} s'appuyant sur les questions 1 et 4. Lequel vous semble le plus précis des deux pour n appels du générateur aléatoire (justifier) ?
7. Soit $1 \leq \alpha \leq 3$. On considère désormais que f appartient à la famille de densités f_α définies sur l'intervalle $(0, 1)$ de la manière suivante

$$\forall v \in (0, 1), \quad f_\alpha(v) = c_\alpha v^\alpha (1-v).$$

Calculer la constante c_α . À quel choix de α correspond l'algorithme de calcul de \mathcal{I} le plus précis ? La précision est-elle supérieure à celle de l'algorithme s'appuyant sur la question 1 ?

Probabilités Appliquées 2
Mars 2009

Durée 2h00 - Documents et calculatrices non autorisés.

I. Exercice

(3pt) Soit un point pris au hasard uniformément dans le disque unité et (R, θ) ses coordonnées polaires. On rappelle que R et θ sont des variables indépendantes et que la densité de R est donnée par $f(r) = 2r\mathbf{1}_{(0,1)}(r)$, pour tout $r \in \mathbb{R}$. On pose

$$W = \frac{1}{R} - 1.$$

Calculer le jacobien de la transformation suivante

$$\begin{cases} Z = W \cos \theta \\ T = W \sin \theta \end{cases}$$

et en déduire la densité du couple (Z, T) (On ne cherchera pas à dériver la transformation inverse). Les variables (Z, T) sont elles indépendantes ?

II. Problème

1. (1pt) Rappeler le principe de simulation d'un couple de variable aléatoires (X, Y) de densité $f(x, y)$.

2. (2pt) On considère un couple (X, Y) de densité définie par

$$\forall (x, y) \in \mathbb{R}^2, \quad f(x, y) = \begin{cases} 2e^{-x-y} & \text{si } 0 < x < y, \\ 0 & \text{sinon.} \end{cases}$$

Calculer les densités des lois marginales et des lois conditionnelles du couple (X, Y) .

3. (2pt) Montrer que la loi conditionnelle de Y sachant $X = x$ ($x > 0$) peut se représenter comme la loi de la somme $x + W$, où W est une variable de loi exponentielle de paramètre 1.

4. (3pt) Ecrire un algorithme de simulation du couple (X, Y) faisant explicitement appel à un changement de variables. Prouver la validité de cet algorithme.

5. (1pt) Soit `alea` un générateur aléatoire de loi uniforme. On considère l'algorithme suivant

```
Repete
U <- -log(alea) ;
V <- -log(alea) ;
Jusqu'a (U < V) ;
X <- U ; Y <- V ;
```

Argumenter (sans calcul) du fait que la loi du couple (X, Y) en sortie de cet algorithme est identique à la loi du couple $(\min(U, V), \max(U, V))$, où U, V sont des variables indépendantes et de loi exponentielle de paramètre 1.

6. (1pt) En déduire que la loi de X en sortie de l'algorithme précédent est la loi exponentielle de paramètre 2.

7. (2pt) Soit $x > 0$ et soit $t > x$. En sortie de l'algorithme précédent, justifier que l'on a

$$P(Y \leq t \mid X = x) = \frac{P(x < V \leq t)}{P(V > x)},$$

et calculer la densité de la loi conditionnelle de Y sachant $X = x$.

8. (1pt) Démontrer que le couple (X, Y) en sortie de l'algorithme précédent admet $f(x, y)$ pour densité jointe.

9. (2pt) Soit $x > 0$. En utilisant la question 3, montrer que $E[Y \mid X = x] = x + 1$. En déduire que $E[XY \mid X = x] = x^2 + x$.

10. (2pt) Déduire de la question précédente l'espérance $E[Y]$ et la covariance $\text{cov}(X, Y)$.



PROBABILITES 1

Examen final

Date de l'épreuve : 28/01/2009

Horaire : de 9h à 11h (Durée : 2 heures)

Documents et calculatrices interdits

Ce sujet comporte 2 pages d'énoncé

Merci de commencer par **noter votre groupe de TD** sur la 1^{ère} page de votre copie pour faciliter le tri.
Ce devoir est composé de 3 exercices indépendants que vous pouvez traiter dans n'importe quel ordre.
Le barème est donné à titre indicatif, et est susceptible d'être modifié.

Il sera grandement tenu compte de la présentation et de la qualité de la rédaction (justification des réponses) dans la notation. Tout résultat manifestement erroné et non signalé (comme une variance négative, ou une probabilité supérieure à 1) entraînera la note de 0 à la question toute entière.

EXERCICE 1 (6 pts)

Soient X et Y deux variables aléatoires indépendantes suivant la même loi géométrique de paramètre p. On note $q = 1-p$.

On pose : $M = \min(X, Y)$ et $N = \max(X, Y)$.

1. Déterminer la loi de probabilité de la variable aléatoire M. Pour ce faire, on pourra calculer $P(M > k)$.
Déterminer la fonction génératrice de M. En déduire l'espérance et la variance de M.
2. En remarquant que $X + Y = M + N$, calculer l'espérance de N.
3. Calculer la covariance de M et N. (On pourra avantageusement remarquer que $MN = XY\dots$)
4. Déterminer la loi de probabilité de la variable aléatoire N. (On pourra décomposer l'événement $\{N = k\}$ en fonction d'événements liés au positionnement de X et Y par rapport à k.)

Un petit rappel mathématique pour cet exercice sur la somme de termes consécutifs d'une suite géométrique:
$$\sum_{j=1}^k q^{j-1} = \frac{1-q^k}{1-q}$$

EXERCICE 2 (7 pts)

Soit X une variable aléatoire réelle continue de support $[-\alpha ; \alpha]$ ayant pour densité $f_X(x) = \xi(\alpha - |x|)$, où ξ et α sont des réels.

Soient U et V des variables aléatoires réelles indépendantes qui suivent une loi uniforme sur $[0,1]$.

On suppose U indépendante de X .

1. Représenter la densité de X , calculer ξ .
2. Calculer la fonction de répartition de X , et écrire un algorithme de simulation de X par inversion.
3. Déterminer la loi de $Y = \alpha[2.\min(U,V) - 1]$.
4. Soit $Z = \mathbf{1}_{\left\{U < \frac{1}{3}\right\}} X + \mathbf{1}_{\left\{U \geq \frac{1}{3}\right\}} \alpha V$.
 - a. Calculer les espérances de X et de X^2 , puis en déduire celles de Z et de Z^2 .
 - b. Déterminer la densité de probabilité de Z et écrire un algorithme de simulation de Z par mélange.

EXERCICE 3 (7 pts)

Soit X une variable aléatoire réelle dont la loi admet pour densité $f_X(x) = \frac{\lambda}{2} \exp(-\lambda|x|)$ où λ est un réel positif.

1. Calculer l'espérance, la variance, ainsi que la fonction caractéristique de X .

Soient U et V deux variables aléatoires réelles indépendantes qui suivent la loi exponentielle de paramètre λ (dont on rappelle que la densité vaut $\lambda \exp(-\lambda x)$ pour x positif).

2. Donner l'espérance, la variance et la fonction caractéristique de U .
3. Déterminer la loi de la variable $T = -U$. Calculer la fonction caractéristique de T .
4. Déterminer la loi de $Y = U - V$ en utilisant la fonction caractéristique.
5. Se servir de la question 4 pour retrouver $E(X)$ et $\text{Var}(X)$.

EXAMEN du 25 mars 2008. Durée: 3h. 3 pages numérotées.
Documents autorisés, mais pas les feuilles avec les exercices résolus.

Veuillez noter sur votre copie le nom de votre enseignant : BIENIA ou CANI ou NADDEF ou SZIGETI

EXERCICE 1 [5 pts]:

Dans un atelier 2 pièces de la carrosserie, notés C_1 et C_2 , sont soumis aux 4 opérations (notées M_1 , M_2 , M_3 et M_4): découpage, dégraissage, traitement anti-corrosion, emballage. Pour chaque opération on dispose d'une seule machine spécialisée. Ces deux pièces peuvent être traitées simultanément sur différentes machines mais chaque machine peut traiter une seule pièce à la fois et ainsi, pour chaque machine, il faut choisir soit C_1 avant C_2 soit C_2 avant C_1 . Pour chaque pièce, les 4 opérations doivent être obligatoirement réalisées dans l'ordre cité. Le tableau ci-contre indique les durées respectives en minutes. Chaque opération commencée, doit être complètement achevée sans pouvoir être interrompue. On doit chercher l'ordre de passage sur chaque machine pour minimiser la durée totale – problème classifié comme **ordonnancement complexe**.

	C_1	C_2
M_1	2	7
M_2	7	2
M_3	5	2
M_4	2	5

- a) Supposons d'abord que chaque machine traite toujours la pièce C_1 avant C_2 . Montrer, en choisissant bien les tâches élémentaires, que ce problème peut être considéré comme ordonnancement simple. Utiliser la méthode **potentiels-tâches** pour trouver la durée totale minimale. Donner la liste les tâches critiques et les marges des autres tâches. [2 pts]
- b) Supposons que l'on peut remplacer une seule machine (parmi les 4) par une machine capable de réaliser l'opération plus rapidement, avec la nouvelle durée égale à une minute par pièce. Quelle machine doit-on choisir pour diminuer le plus fortement la durée totale? Donner la nouvelle durée totale minimum et les nouvelles marges pour le traitement des pièces C_1 et C_2 par la nouvelle machine. [1 pt]
- c) On revient aux durées initiales. Pour résoudre le problème d'ordonnancements complexes posé (trouver l'ordre de passage sur chaque machine pour minimiser la durée totale), on n'est pas obligé d'étudier toutes les 2^4 possibilités, car on peut supposer un ordre identique sur les 2 premières machines et aussi un ordre identique sur les 2 dernières (on doit admettre ce résultat). En étudiant les 4 problèmes d'ordonnancement simple, déterminer dans quel ordre doit-on traiter les pièces C_1 et C_2 sur chaque machine, pour minimiser la durée totale. Présenter le résultat optimal sur un diagramme de Gantt. [2 pts]

EXERCICE 2 [4 pts]:

On veut acheter des rayonnages pour ranger des livres. Les livres sont de n hauteurs $0 < H_1 < H_2 < \dots < H_n$ et on connaît le métrage linéaire L_i des livres de hauteur H_i . Le rayonnage pour les livres hauts doivent être plus épais (donc plus cher) que ceux des petits livres, ceci à cause du poids. Le coût de x_i mètres de rayonnage pour livres de hauteur H_i est $F_i + C_i x_i$, où F_i est un coût fixe indépendant de la quantité commandée et C_i le coût du mètre linéaire.

On n'est pas obligé d'acheter du rayonnage adapté à chaque hauteur car on peut mettre des livres de hauteur $H_j \leq H_i$ sur un rayonnage prévu pour des livres de hauteur H_i .

- a) Modéliser le problème de trouver quelle quantité de rayonnage acheter de chaque type qui minimise le coût total. Bien justifier votre modèle. La note sera pour bonne partie sur cette justification.
- b) Exemple numérique : supposons que tous les livres ont 2 cm d'épaisseur et il y a 500 livres de hauteur $H_1=15$ cm, 2000 livres de hauteur $H_2=20$ cm, 4500 livres de hauteur $H_3=25$ cm et 3000 livres de hauteur $H_4=30$ cm. Fixons $F_1=F_2=60$; $F_3=F_4=70$; $C_1=6$; $C_2=8$; $C_3=9$; $C_4=10$.

Résoudre l'exemple numérique et expliquer le rangement optimal.

EXAMEN du 25 mars 2008. Durée: 3h. 3 pages numérotées.

EXERCICE 3 [4 pts]:

Soit $G(X, A)$ un graphe orienté. Soit $p: X \rightarrow \mathbb{R}$ une fonction qui à un sommet x de X associe une valeur réelle $p(x)$.

- a) Montrer qu'une condition nécessaire pour que $p(x)$ soit la plus courte distance d'un sommet a à x est que, pour tout y de X tel que (y, x) est un arc, alors $p(x) \leq p(y) + l(y, x)$, où $l(y, x)$ est la longueur de l'arc (y, x) .
- b) Montrer que, si de plus $p(a)=0$ et, pour tout x de X , il existe un chemin de a à x de longueur $p(x)$, alors effectivement les $p(x)$ sont les plus courtes distances de a à x pour tout x . On a alors une condition nécessaire et suffisante.

CONSEIL : Considérez un chemin quelconque reliant a à x et montrez, en utilisant le fait que les $p(\cdot)$ vérifient la condition suffisante, que sa longueur est au moins $p(x)$.

EXERCICE 4 [4 pts]:

Un hameau de montagne situé au terminus d'une route est composé de 11 chalets, reliés entre eux et au parking par des chemins piétonniers. Pour réduire la facture électrique, le maire de la commune propose de n'éclairer qu'une partie de ces chemins, par des réverbères régulièrement espacés. Il est entendu avec les habitants, qu'ils disposeront chacun d'un réverbère allumé devant leur chalet, qu'il y en aura un au parking, et qu'ils pourront se rendre mutuellement visite, en ne passant que par des chemins éclairés.

Le plan du village, donné sur la feuille ci-jointe, porte le nombre de réverbères nécessaires le long de chaque chemin (en plus de ceux situés devant les chalets).

- a) Trouver la solution d'éclairage la moins coûteuse pour la commune (c'est-à-dire le nombre de réverbères à allumer et leur plan de placement), en utilisant un algorithme vu en cours que l'on précisera. [1 pts]

Le maire habite le chalet K et souhaite pouvoir emprunter le chemin le plus court pour rendre visite de nuit à sa fiancée, qui réside dans le chalet F.

- b) Calculer ce plus court chemin, en précisant l'algorithme utilisé, dont on détaillera les étapes. Quelle condition d'arrêt peut-on ajouter sur la boucle principale ? [1 pts]
- c) Proposer une méthode pour modifier, sans tout recalculer, la solution trouvée en a) de manière à ce que le trajet nocturne du maire soit bien éclairé. Combien de réverbères supplémentaires faudra t'il allumer ? [2 pts]

EXERCICE 5 [4 pts]:

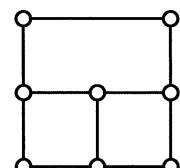
- a) Montrer que chaque graphe orienté $G(X, A)$ se décompose en deux graphes partiels sans circuits, c'est-à-dire qu'il existe une partition de l'ensemble d'arcs $A = A_1 \cup A_2$ ($A_1 \cap A_2 = \emptyset$), telle que les graphes $G(X, A_1)$ et $G(X, A_2)$ sont sans circuit.

CONSEIL : Numéroter les sommets de façon arbitraire et penser au tri topologique.

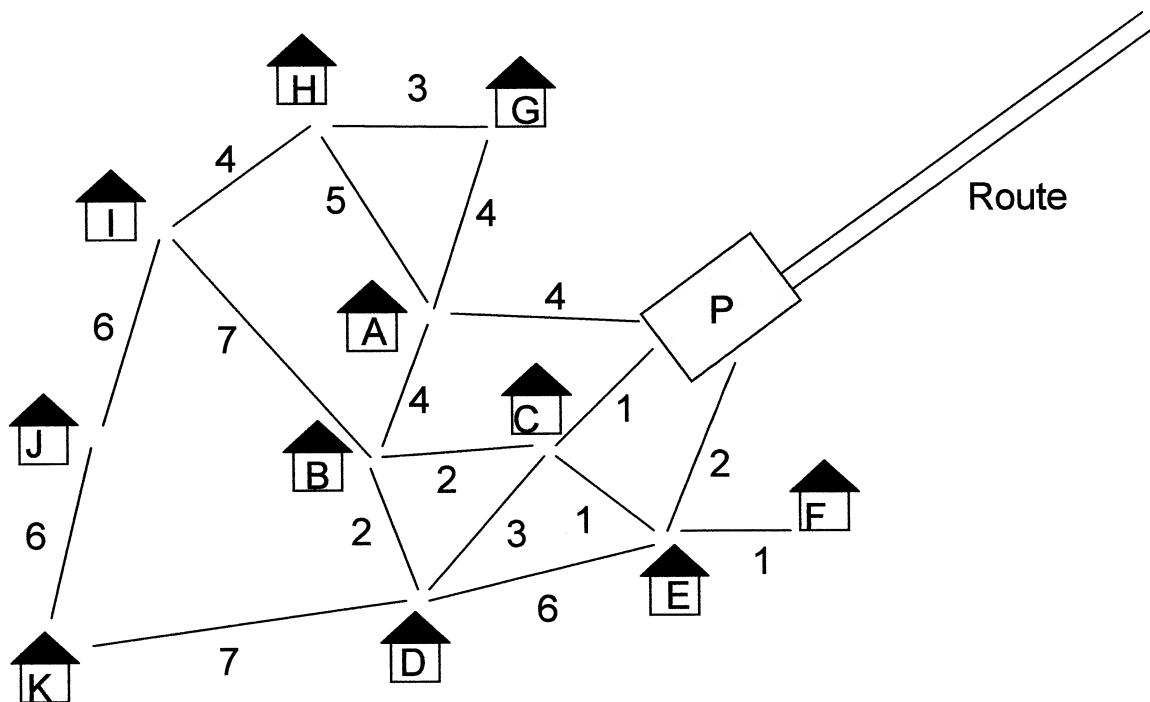
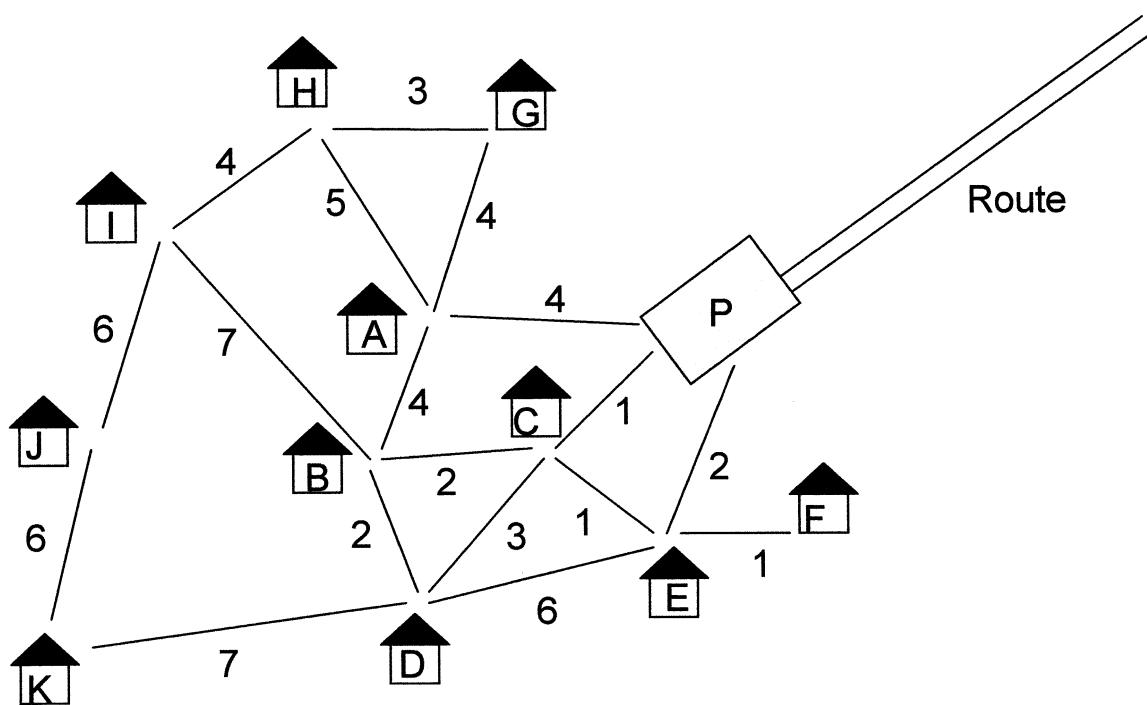
- b) Supposons qu'on peut supprimer k arcs d'un graphe orienté G de telle sorte que le graphe obtenu ne contienne pas de circuit. Montrer qu'on peut éliminer les circuits de G en changeant la direction d'au plus k arcs de G .

EXERCICE 6 [2 pts]:

- a) Existe-t-il une chaîne eulérienne dans le graphe ci-contre ? Pourquoi ? [0,5 pt]
- b) Trouver le nombre minimum d'arêtes d'une chaîne fermée, qui utilise chaque arête au moins une fois ? Justifier votre réponse. [1,5 pt]



Nom, prénom :



EXAMEN du 20 mai 2009. Durée: 3h. 3 pages numérotées.
Polycopié du cours et les documents manuscrits autorisés.

Veuillez noter sur votre copie le nom de votre enseignant : BIENIA ou CANI ou NADDEF ou SZIGETI

EXERCICE 1 [6 pts]:

Une entreprise fabrique deux produits **P1** et **P2**. La production nécessite du temps de travail : **45 minutes** de main d'œuvre par unité de **P1** et **30 minutes** par unité de **P2**; du temps-machine : **90 minutes** par unité de **P1** et **48 minutes** par unité de **P2** et de la matière première (pièces préfabriquées) fournie par un sous-traitant sous la forme de coffrets standard. La fabrication d'une unité de **P1** nécessite **3** coffrets standards et **P2 2** coffrets standard. Les produits sont vendus **30 €** une unité de **P1** et **20 €** une unité de **P2**.

La main d'œuvre est fournie par **2** ouvriers employés à plein temps (**35** heures par semaine chacun) ce qui donne un coût fixe pour l'entreprise. Ces personnes peuvent effectuer des heures supplémentaires qui sont payées **15 €** l'heure. Chaque semaine, la disponibilité en temps-machine est de **320 h**. La matière première peut être achetée au prix de **3 €** par coffret, dans la limite d'au plus **400** coffrets standard par semaine.

En l'absence de publicité, la demande hebdomadaire du produit **P1** serait de **50** unités, celle de **P2** de **65** unités; mais on peut réaliser de la publicité pour développer les ventes : chaque euro dépensé en publicité sur **P1** augmente la demande hebdomadaire de **P1** de **10** unités et chaque euro dépensé en publicité sur **P2** augmente la demande hebdomadaire de **P2** de **20** unités. Les frais de publicité ne doivent pas dépasser **200 €** par semaine. Les quantités de **P1** et **P2** fabriquées doivent rester inférieures ou égales à la demande réelle (compte tenu de la publicité). L'entreprise désire organiser sa production de manière à maximiser son bénéfice net. Ce problème a été formulé comme un programme linéaire et résolu par la méthode du simplexe. Voici le premier et le dernier tableau (Attention ! Le tableau utilise les coefficients opposés à ceux de la fonction-objectif à maximiser, comme dans le poly) :

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}		
0.75	0.5	0	0	-1	1	0	0	0	0	0	70	
1.5	0.8	0	0	0	0	1	0	0	0	0	320	
3	2	0	0	0	0	0	1	0	0	0	400	
1	0	-10	0	0	0	0	0	1	0	0	50	
0	1	0	-20	0	0	0	0	0	1	0	65	
0	0	1	1	0	0	0	0	0	0	1	200	
-21	-14	1	1	15	0	0	0	0	0	0	0	
												0
												2347
0	0	0.75	1	0	0	0	0.025	-0.075	-0.05	0	3	
0	0	3	0	0	0	1	-0.4	-0.3	0	0	145	
0	0	0	0	1	-1	0	0.25	0	0	0	30	
1	0	-10	0	0	0	0	0	1	0	0	50	
0	1	15	0	0	0	0	0.5	-1.5	0	0	125	
0	0	0.25	0	0	0	0	-0.025	0.075	0.05	1	197	
0	0	0.25	0	0	15	0	3.225	0.075	0.05	0	2347	

- a) Ecrire le programme primal sous forme canonique. Interpréter les 11 variables et les 6 contraintes primaires présentées dans les tableaux. Quelle est la base optimale ? Indiquer sa matrice B et son inverse B^{-1} . Donner la solution optimale et le profit correspondant.
- b) Ecrire le programme dual, donner sa solution optimale et l'interprétation de chaque variable duale.
- c) La disponibilité en temps-machine baisse de 320 à 200. La solution optimale est-elle modifiée ? Quelle baisse de cette disponibilité peut-on accepter sans que la solution optimale soit changée ?
- d) On peut acheter (vendre) une quantité marginale de coffrets standard au prix de 4 € par coffret. Conseilleriez-vous d'acheter (vendre) quelques coffrets ? Quelles sont les quantités limites quand on achète et quand on vend ?

EXERCICE 2 [4 pts]:

Voici le système linéaire :

$$\begin{aligned}x_1 + x_2 &= -1 \\2x_1 - x_2 &= 4 \\x_1 &\geq 0.\end{aligned}$$

où la variable x_2 est non astreinte.

Mettre ce système sous forme standard et, en utilisant la phase I du simplexe, trouver une solution de base réalisable. Donner explicitement cette solution.

EXERCICE 3 [3 pts]

Considérons le programme linéaire suivant. Dire, sans résoudre ce programme linéaire, si la solution proposée est optimale ou non. Bien détailler toutes les étapes du processus de preuve.

$$\begin{array}{ll}\text{maximiser: } z = & 7x_1 + 6x_2 + 5x_3 - 2x_4 + 3x_5 \\ \text{sous: } & x_1 + 3x_2 + 5x_3 - 2x_4 + 2x_5 \leq 4 \\ & 4x_1 + 2x_2 - 2x_3 + x_4 + x_5 \leq 3 \\ & 2x_1 + 4x_2 + 4x_3 - 2x_4 + 5x_5 \leq 5 \\ & 3x_1 + x_2 + 2x_3 - x_4 - 2x_5 \leq 1 \\ & x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0.\end{array}$$

Solution proposée : $\underline{x}_1=0, \underline{x}_2=4/3, \underline{x}_3=2/3, \underline{x}_4=5/3, \underline{x}_5=0$.

EXERCICE 4 [4 pts]: Gestion de production:

Une entreprise connaît la demande pour les N périodes à venir. Soient $d_1, \dots, d_i, \dots, d_N$ ces demandes. Les coûts de production varient selon la période et seront notés c_i par unité pour la période i , $i=1, \dots, N$. Cette variation est par exemple due au coût variable de l'énergie. On peut stocker d'une période sur l'autre, le coût est de s par période et par unité. Les unités produites et livrées ne sont pas stockées. En d'autres termes, seuls les produits présents en fin de période sont stockés. La capacité de stockage est limitée à S par période (capacité de l'entrepôt). On suppose qu'il n'y a pas de limite de capacité de production. On a un stock initial I_0 et on veut terminer avec un stock I_N .

Le problème est de trouver le niveau de production de chaque période qui minimise le coût total tout en satisfaisant la demande.

- a) Montrer que si les coûts de production sont identiques pour chaque période, alors la solution optimale du problème est triviale.
- b) En associant à la production de chaque période une variable, de même pour le stock à la fin de chaque période, écrire un programme linéaire qui donne le niveau de production pour chaque période.
- c) Cette fois-ci on utilise comme variables $x_{ij} =$ quantité produite au cours de la période i pour satisfaire de la demande de la période j . Ecrire le programme linéaire avec ces variables. On supposera pour cette question $I_0=0$.

Remarque: En l'absence de capacités de stockage, le deuxième programme linéaire, qui semble a priori moins bon car il possède plus de variables et est moins intuitif, a toujours une solution optimale entière, ce qui peut être intéressant quand il s'agit de production de voitures etc...

EXERCICE 5 [5 pts]:

Rappelons les définitions d'un couplage et d'un ensemble transversal et le théorème de König :

Un ensemble C d'arêtes de G est un **couplage** si pour tout sommet v de G au plus une arête de C est incidente à v .

Un ensemble T de sommets de G est un **transversal** si pour toute arête e de G au moins une des deux extrémités de e appartient à T .

THEOREME (König) : Pour tout graphe biparti G , le cardinal maximum d'un couplage de G est égal au cardinal minimum d'un ensemble transversal de G .

- a) Formuler le problème du couplage de cardinal maximum dans un graphe biparti G comme un programme linéaire en nombres entiers. Bien justifier votre modèle.

Dans la suite, A désigne la matrice d'incidence d'un graphe biparti $G=(V;E)$; $x=(..., x_e, ...)$ le vecteur avec des variables indexées par les arêtes du graphe G et $y=(..., y_v, ...)$ le vecteur avec des variables indexées par les sommets du graphe G , \mathbb{I} le vecteur dont toutes les coordonnées sont 1.

- b) Montrer que toute solution optimale entière du programme linéaire :

$$\begin{aligned} \text{maximiser : } z &= \sum_{e \in E} x_e \\ [\mathbf{P}] \quad \text{sous : } Ax &\leq \mathbb{I} \\ &x \geq \mathbf{0}, \end{aligned}$$

est une solution optimale du programme linéaire en nombres entiers

$$\begin{aligned} \text{maximiser : } z &= \sum_{e \in E} x_e \\ \text{sous : } Ax &\leq \mathbb{I} \\ x_e &\in \{0, 1\} \end{aligned}$$

- c) Ecrire le dual $[\mathbf{D}]$ de $[\mathbf{P}]$. En utilisant la structure de la matrice A , montrer que

$$\begin{aligned} \text{minimiser : } w &= \sum_{v \in V} y_v \\ [\mathbf{D}] \quad \text{sous : } y_u + y_v &\geq 1 \text{ pour toute arête } uv \text{ de } G \\ &y \geq \mathbf{0}. \end{aligned}$$

- d) Admettons que les programmes linéaires $[\mathbf{P}]$ et $[\mathbf{D}]$ possèdent toujours des solutions optimales qui sont entières. En déduire le théorème de König en utilisant le théorème fondamental de la dualité.
- e) Traduire les conditions des écarts complémentaires pour $[\mathbf{P}]$ et $[\mathbf{D}]$ pour le graphe biparti G .

EXERCICE 6 [4 pts]

Soit le jeu où vous et votre adversaire choisissez chacun un entier de 1 à 100. Si votre nombre x est strictement inférieur à y choisi par votre adversaire alors vous gagnez une mise, sauf le cas $x=y-1$ où c'est votre adversaire qui gagne. Si, au contraire, votre nombre x est strictement supérieur à y choisi par votre adversaire alors vous perdez, sauf dans le cas où $x=y+1$ dans lequel le vainqueur c'est vous. Si $x=y$ alors il y a match nul.

- a) Trouver la matrice de gains. Sans chercher la stratégie optimale démontrer que la valeur du jeu est nulle. Utiliser une propriété évidente de la matrice obtenue.
- b) Ecrire les deux programmes duals qui correspondent à la recherche des stratégies optimales des deux joueurs. Démontrer, en utilisant la dualité, que les deux joueurs ont les mêmes stratégies optimales : choisir 1, 2 ou 3 avec les mêmes fréquences et de ne jamais choisir 4, 5, ..., 100.

Théorie des langages 1

Durée : 2h

Documents : tous documents autorisés

Le barème est indicatif (total sur 20 points).

Avant de répondre aux questions, merci d'en lire attentivement le libellé : la **forme** attendue pour la réponse est précisée.

1 Schéma d'induction et induction structurelle (12 points)

Soit $V = \{0, 1, R\}$ un vocabulaire. Le schéma d'induction suivant définit le langage $E \subseteq V^*$:

- **base** : $0.R.1 \in E$
- **règle 1 (R_1)** : si $u.R.v \in E$ alors $0.u.R.0.v \in E$
- **règle 2 (R_2)** : si $u.R.v \in E$ alors $1.u.R.1.v \in E$
- **règle 3 (R_3)** : si $u.R.v \in E$ alors $u.1.R.v.0 \in E$

A noter : . représente la concaténation et R est un élément du vocabulaire.

On peut constater alors que tous les mots de E sont de la forme : $u.R.v$ avec $u, v \in \{0, 1\}^*$. Cette propriété est **admise**.

Question 1 (1 point) Donner 5 mots de E . Il est conseillé d'utiliser au moins une fois chaque règle.

On considère maintenant que pour $u.R.v \in E$, u et v sont l'écriture binaire de nombres. On note $val(u)$ et $val(v)$ la valeur de ces nombres.

Une expression possible pour $val(x)$ avec $x \in \{0, 1\}^*$ est donnée par :

$$\text{si } x = x_n \dots x_1.x_0 \text{ avec } x_i \in \{0, 1\} \text{ alors } val(x) = \sum_{i=0}^n 2^i x_i$$

Question 2 (2 points) Expliquer en quelques mots pourquoi pour tout mot $w = u.R.v$ de E , on a : $|u| = |v|$.

Question 3 (4 point) Prouver que pour tout mot $u.R.v$ de E , on a : $val(u) + 1 = val(v)$.
On attend une preuve rédigée.

Question 4 (5 points) Prouver que

$$E = \{w \in V^* \text{ tq } w = u.R.v \text{ et } |u| = |v| \text{ et } val(v) = val(u) + 1\}$$

On attend une preuve rédigée.

2 Induction bien fondée (8 points)

On s'intéresse à la fonction $f : \mathbb{N} \rightarrow \mathbb{N}$ suivante :

$$\begin{aligned} f(n) &= f(f(n+6)) \text{ si } n \leq 3000 \\ &= n-5 \text{ sinon} \end{aligned}$$

On veut prouver qu'elle est bien définie, c'est-à-dire qu'elle s'arrête en un nombre fini de pas et qu'elle vaut 2996 si $n \leq 3000$ et $n-5$ sinon. Pour cela, on définit l'ordre \leq sur \mathbb{N} ; m est *plus petit que* n selon \leq s'écrit :

$$m \leq n \iff (n \leq m \text{ et } n \leq 3000) \text{ ou } (n = m \text{ et } n \geq 3000)$$

On pourra noter \lhd l'ordre strict associé :

$$m \lhd n \iff m \leq n \text{ et } m \neq n$$

Question 5 (1 point) Ordonner, pour l'ordre \leq , sur un dessin, les valeurs : 0, 2998, 2999, 3000, 3001, 3002 et 3003.

Ne pas justifier. On pourra omettre les relations qui se déduisent par transitivité. Par exemple si $a \lhd b \lhd c$, on pourra dessiner

$$a \xrightarrow{\lhd} b \xrightarrow{\lhd} c$$

Question 6 (1 point) Quels sont les minimaux de cet ordre ?

Ne pas justifier.

Question 7 (2 point) On admet que la relation \leq est un ordre. Prouver que c'est un ordre bien fondé.

Question 8 (1 point) Calculer $f(3001), f(3000), f(2999)$.

Question 9 (3 point) Prouver par induction bien fondée que :

$$\text{pour tout } n \in \mathbb{N}, f(n) \text{ est bien défini et vaut } f(n) = \begin{cases} 2996 & \text{si } n \leq 3000 \\ n-5 & \text{sinon} \end{cases}$$

On attend une **preuve rédigée**.

Indication : dans le pas d'induction, faire deux cas, l'un pour $n+6 > 3000$ et l'autre pour $n+6 \leq 3000$.

Théorie des langages 1

Durée : 3h

Documents : tous documents autorisés

Forme des réponses : toute réponse doit être justifiée rapidement hormis précision contraire. Il est précisé explicitement (“montrer que, prouver”) quand une preuve doit être fournie.

Le barème est indicatif (total sur 22 points).

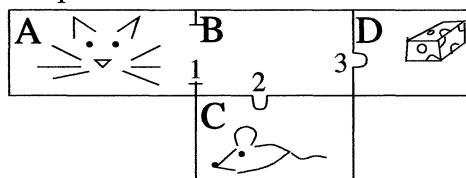
Le sujet est composé de deux problèmes indépendants.

1 Problème de modélisation avec des automates (7 points)

Jeu du chat et de la souris : une souris et un chat se déplacent dans une maison. Le chat cherche à manger la souris ; la souris cherche à aller manger le fromage sans se faire manger par le chat.

1.1 Exemple de jeu

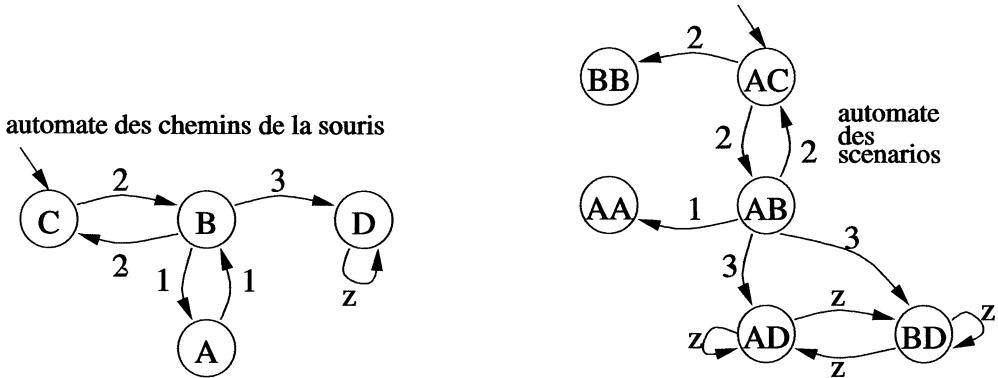
Le plateau de jeu (plan des pièces) est donné ci-dessous (4 pièces étiquetées A, B, C, D). La position initiale est celle du dessin (chat en A , souris en C et fromage en D). Les petites portes (trous de souris, numérotés 2 et 3) sont réservées à la souris, les grandes portes (une grande porte numérotée 1) peuvent être utilisées par le chat et la souris.



Le jeu se joue au tour par tour : à chaque tour, chaque animal effectue un mouvement (franchissement d'une porte, noté par le numéro de porte) ou choisit s'il en a le droit de ne rien faire (mouvement 'z'). Le chat peut ne pas bouger à n'importe quel tour de jeu. La souris ne s'arrête de bouger qu'une fois qu'elle a atteint le fromage.

Un chemin pour la souris est un mot du vocabulaire $V = \{1, 2, 3, z\}$. Par exemple, le mot 2.1.1 mène la souris de son état initial, la pièce C à la pièce B . L'automate des chemins de la souris est donnée ci-dessous.

L'automate des scénarios représente toutes les évolutions possibles du jeu en regardant les mouvements de la souris, quels que soient les mouvements possibles du chat. Quand le chat a mangé la souris (chat et souris dans la même pièce), le jeu s'arrête. Les états sont composés de deux lettres : la première donne la position du chat et la deuxième celle de la souris. Les transitions représentent les mouvements de la souris.



Question 1 *FROM et CHAT* (1 point)

- L'automate des chemins de la souris n'a pas d'états terminaux. On veut que cet automate représente le langage des mots qui mènent la souris au fromage. On nomme ce langage *FROM*. Préciser les états terminaux de l'automate et donner une expression régulière égale à *FROM*.
- L'automate des scénarios n'a pas d'états terminaux. On veut que cet automate représente le langage des scénarios gagnants pour le chat. On appelle ce langage *CHAT*. Que vaut *CHAT* ?

Il est évident dans ce jeu que (si le chat joue correctement) la souris ne peut pas gagner. Cela s'explique car l'ensemble des chemins de la souris qui mènent au fromage ont pour préfixe 2 et 2 est un mot de *CHAT*. On va expliciter cette propriété et trouver comment calculer la solution à un tel jeu si le plateau est moins trivial.

1.2 Transformations d'automates

Les transformations suivantes sont des outils nécessaires à la construction de la solution du jeu

Question 2 Préfixe (2 points) Soit L un langage sur V . On définit le langage des préfixes de L par :

$$\text{prefix}(L) = \{u \in V^* \text{ tel que } \exists v \in V^*, u.v \in L\}$$

On suppose que L est un langage régulier et que $A = \langle Q, V, \delta, i, F \rangle$ est un automate fini de langage L : $L(A) = L$. Proposer une construction permettant de construire un automate de langage $\text{prefix}(L)$. Prouver que votre construction construit bien un automate reconnaissant $\text{prefix}(L)$.

Question 3 Rallonge (1 point) Soit L un langage sur V . On définit le langage rallonge de L par :

$$\text{rallonge}(L) = \{u \in V^* \text{ tel que } \exists x \in L, \exists y \in V^*, x.y = u\}$$

On suppose que L est un langage régulier. Donner une expression régulière représentant $\text{rallonge}(L)$. Si $A = \langle Q, V, \delta, i, F \rangle$ est un automate fini de langage L ($L(A) = L$), proposer une construction permettant de construire un automate de langage $\text{rallonge}(L)$.

1.3 Une solution au problème

V est le vocabulaire des chemins de la souris (i.e. les numéros des portes que la souris peut emprunter). On suppose qu'on connaît les langages sur V^* :

- $FROM$ qui représente l'ensemble des chemins de la souris qui mènent au fromage
 - $CHAT$ qui représente l'ensemble des scénarios qui permettent au chat de gagner.
- On s'intéresse au langage

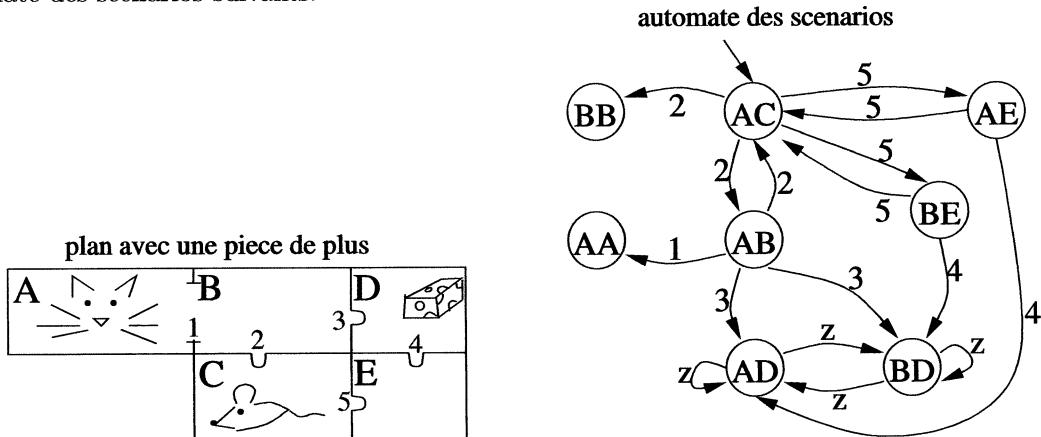
$$SOL = FROM \setminus rallonge(prefix(FROM) \cap CHAT)$$

NB : $A \setminus B = A \cap \overline{B}$.

Question 4 Une solution !! (1 point) Montrer que le langage SOL représente l'ensemble des chemins de la souris qui mènent au fromage sans qu'elle puisse jamais se faire manger par le chat, quels que soient les mouvements du chat.

1.4 Application avec une pièce de plus

On applique maintenant nos résultats à un plateau avec une pièce de plus. Cela conduit à l'automate des scénarios suivants.



Question 5 Déterminisation (0.5point) Déterminiser l'automate des scénarios. On note l'automate résultant DET .

Question 6 Calcul de SOL (1.5points) A partir de l'automate DET , donner successivement les automates représentant les langages suivant.

- Les langages $CHAT$ et $FROM$: comme les chemins de la souris qu'on considère sont tous des scénarios du jeu, on peut prendre pour automate représentant le langage $FROM$ (respectivement $CHAT$) l'automate DET , en précisant l'ensemble des états terminaux. Donner, pour $CHAT$, puis pour $FROM$ leurs états terminaux respectifs.
- Les langages $prefix(FROM)$ puis $prefix(FROM) \cap CHAT$ puis $rallonge(prefix(FROM) \cap CHAT)$.
- Le langage SOL .

Expliquer et justifier rapidement si nécessaire les étapes de vos calculs.

2 Langages réguliers de mots infinis (15 points)

Le but de ce problème est d'étendre la notion de langages réguliers aux mots infinis. On parle alors de langages ω -réguliers. Un mot infini est une séquence infinie de lettres : $(a_i)_{i \geq 0} = a_0a_1\dots a_i\dots$.

Soit V un alphabet fini, V^* est l'ensemble des mots finis sur V et on note V^ω l'ensemble des mots infinis sur V .

2.1 Opérations

On définit alors comme d'habitude des opérations sur ces objets. Le $+$ représente maintenant l'union de langages qu'ils soient sur des mots finis ou infinis. Le $.$ (concaténation) est étendu à des mots infinis (cf. concaténation infinie). L' $*$ n'est pas modifié mais on définit l'opération ω (cf. puissance) qui est le pendant de l' $*$ pour les mots infinis.

La concaténation infinie : si $u \in V^*, v \in V^\omega$ alors $u.v \in V^\omega$ tel que si $u = u_0\dots u_n$ et $v = v_0\dots v_i\dots$ alors $u.v = u_0\dots u_nv_0\dots v_i\dots$. Intuitivement on colle le mot fini u à gauche du mot infini v .

L'opération de puissance : pour $L \subseteq V^*$, on note $L^\omega = \{(u_i)_{i \geq 0}, u_i \in L, u_i \neq \epsilon\}$. Pour $L = \{\epsilon\}$, $L^\omega = \emptyset$. Cette notion de puissance est l'extension de l' $*$ au sens où c'est l'ensemble des concaténations infinies de mots de L . Cette notation est cohérente avec la définition de V^ω .

Question 7 (3 points) Soient $X \subseteq V^*$ et $Y \subseteq V^*$ des langages de mots finis sur l'alphabet V .
Montrer que

- (a) $(XY)^\omega = X(YX)^\omega$
- (b) $XX^\omega = X^\omega$
- (c) $(X + Y)^\omega = (X^*Y)^\omega + (X + Y)^*X^\omega$

Indication : raisonner sur la forme générale des mots, sans faire de preuve par induction.

2.2 Automates pour les mots infinis (automates de Büchi)

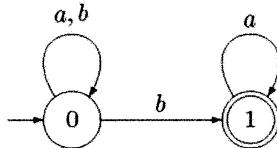
On propose une interprétation des automates finis non déterministes classiques pour pouvoir accepter des mots infinis. En effet, un mot infini ne se terminant jamais, cela n'a plus de sens de dire que l'on accepte un mot si l'on s'arrête sur un état final. En revanche, un chemin étiqueté par un mot infini passe forcément par certains états une infinité de fois (le nombre d'état d'un automate étant fini). On considère alors qu'**un mot infini u est accepté par un automate** si il existe un chemin étiqueté u qui part de l'état initial et passe une infinité de fois par un état final.

Pour un automate $A = < Q, V, \delta, i, F >$. On notera $L_\omega(A)$ le langage de mots infinis reconnu par A ($L_\omega(A) \subseteq V^\omega$). Et on conserve la notation $L(A)$ pour les mots finis reconnus par A avec la définition habituelle ($L(A) \subseteq V^*$).

On ne fait pas d'hypothèse a priori sur A :

- $\delta \subseteq Q \times V \cup \{\epsilon\} \times Q$, autrement dit, A peut comporter des ϵ -transitions, ne pas être déterministe, ...
- $F \subseteq Q$

Question 8 (0.5 point) Soit A_1 l'automate suivant :



Que valent $L(A_1)$ et $L_\omega(A_1)$? Est-ce que $[L(A_1)]^\omega = L_\omega(A_1)$?

Question 9 (0.5 point) Donner un automate A_2 tel que

$$L_\omega(A_2) = \{u \in \{a, b\}^* \text{ tel que } |u|_a = \infty \text{ et } |u|_b = \infty\}$$

Question 10 (2 points)

- (a) Soient $A_1 = < Q_1, V, \delta_1, i_1, F_1 >$ et $A_2 = < Q_2, V, \delta_2, i_2, F_2 >$ des automates finis. Construire un automate fini qui reconnaît le langage $L_\omega(A_1) + L_\omega(A_2)$.
- (b) Soit $L \subseteq V^*$ un langage régulier de mots finis. Soit $A = < Q, V, \delta, i, F >$ un automate fini. Construire un automate fini qui reconnaît le langage $L.L_\omega(A)$.
- (c) Soit $L \subseteq V^*$ un langage régulier de mots finis. Construire un automate fini qui reconnaît le langage L^ω . Prouver cette dernière construction.

Question 11 (3 points) On dit qu'un langage de mots infinis $L \subseteq V^\omega$ est **ω -régulier** s'il est reconnu par un automate fini. Montrer qu'un langage de mots infinis $L \subseteq V^\omega$ est ω -régulier si et seulement si $L = \bigcup_{i=1}^n X_i Y_i^\omega$ avec $n \in \mathbb{N}$ et $X_i, Y_i \in V^*$ des langages de mots finis réguliers.

2.3 Cas déterministe

On se pose la question de savoir si le mode de reconnaissance que nous venons de définir donne le même ensemble de langage pour les automates déterministes et les automates non déterministes.

On définit la **limite d'un langage de mots finis** $L \subseteq V^*$ par

$$\overrightarrow{L} = \{u \in V^\omega \text{ tel que } u = u_1 u_2 \dots u_i \dots \text{ et } u_1 u_2 \dots u_n \in L \text{ pour une infinité de } n\}$$

Autrement dit la limite de L est l'ensemble des mots infinis qui possèdent une infinité de préfixes dans L .

Question 12 (1 point) Montrer que $\overrightarrow{(a+b)^+b}$ est l'ensemble des mots u de $\{a, b\}^\omega$ tels que $|u|_b = \infty$.

Question 13 (2 points) Montrer que le langage $(a+b)^*a^\omega$ n'est pas la limite d'un langage de mots finis.

Question 14 (3 points)

- (a) Soit $A = < Q, V, \delta, i, F >$ un automate *déterministe*. Montrer que $L_\omega(A) = \overrightarrow{L(A)}$.
- (b) Est-ce que la propriété précédente est vraie pour un automate non déterministe. Justifiez votre réponse. Qu'en déduisez-vous ?

Théorie des langages 2

Durée : 3h.

Documents : tous documents autorisés.

Exercice (6 points) Partie Calculabilité

On considère les machines de Turing avec une tête de lecture seulement, MT_l . C'est-à-dire que de telles machines ne peuvent pas écrire sur le ruban. Une machine de Turing avec une tête de lecture est de la forme $M = (Q, V, \delta, q_0, B, F)$ avec :

- Q un ensemble fini d'états
- V , le vocabulaire, un ensemble fini de symboles
- δ la fonction de transition de profil $Q \times V \rightarrow Q \times \{G, D, N\}$
- q_0 l'état initial ($\in Q$)
- B le symbole “blanc” ($\in V$)
- F est l'ensemble d'états finaux ($F \subseteq Q$)

Le fonctionnement d'une telle machine est le même que celui des machines de Turing classiques, sauf que dans la fonction de transition, il n'est pas prévu de remplacer (réécrire) le caractère lu par la tête de lecture. Une chaîne ω est acceptée par une telle machine si et seulement si il existe une dérivation

$$c_0 \Rightarrow c_1 \Rightarrow \dots \Rightarrow c_n \not\Rightarrow$$

telle que c_0 est la configuration initiale $q_0\omega$ et c_n est une configuration terminale de la forme $\alpha_1 q \alpha_2$ avec q un état final ($q \in F$) et α_1 et α_2 étant des chaînes sur V ($\alpha_1, \alpha_2 \in V^*$).

▷ **Question 1** (2 points)

Donner une MT_l qui accepte le langage a^*b^* .

▷ **Question 2** (2 points)

Soit M une MT_l quelconque et w une entrée quelconque. Montrer que si M s'arrête pour w alors on peut borner le nombre de transitions effectuées en fonction de la taille de w et du nombre d'états de M .

▷ **Question 3** (2 points)

Soit $M = (Q, V, \delta, q_0, B, F)$ une MT_l et w une chaîne sur V . Le problème “ M accepte w ” est-il décidable ?

Problème (14 points) Langage Hors-contexte et reconnaiseur

On considère le langage **LB** décrit par la grammaire suivante :

```
1 program → bloc
2 bloc → decl begin suite-inst end
3 suite-inst → suite-inst inst ;
4 suite-inst → inst ;
5 inst → bloc
6 inst → idf := exp
7 exp → idf
8 exp → num
9 decl → ε
10 decl → declare suite-idf : integer ;
11 suite-idf → idf , suite-idf
12 suite-idf → idf
```

Les éléments de vocabulaire terminal sont notés en gras.

▷ Question 1 (3 points)

Donner une grammaire LL(1) pour ce langage. On justifiera la réponse en calculant les directeurs de chaque règle. On pourra utiliser des numéros de règles, à condition que la numérotation utilisée soit claire.

▷ Question 2 (3 points)

Ecrire les procédures d'analyse permettant de reconnaître les blocs (non-terminal bloc) et les suite d'instructions (non-terminal suite-inst). On utilisera les conventions du TP (variable mot-cour contenant le mot courant et fonction lire-mot retournant le prochain mot). **On supposera que les procédures permettant de reconnaître les instructions (non terminaux inst et exp) et les déclarations (non terminaux decl et suite-idf) sont déjà écrites.**

▷ Question 3 (3 points)

Le langage **LB** permet d'imbriquer des blocs qui délimitent la durée de vie des variables déclarées dans ces blocs. Soit le programme suivant :

```
1. declare x : integer ;
2. begin      x:= 1 ;
3.           declare y : integer ; begin y:=x ; end ;
4.           declare z, u : integer ; begin u:=x ; z:=u ; end ;
   end
```

Dans l'exemple précédent la variable **y** ne peut être utilisée que dans le bloc qui la déclare, de même que **z** et **u**. En terme de place mémoire on peut donc réutiliser le même emplacement mémoire pour stocker les variables **y** et pour **z** par exemple. On veut calculer la place mémoire maximum nécessaire à un programme. On supposera que les entiers sont codés sur 4 cases mémoire. Donner un calcul d'attributs permettant d'évaluer le nombre de cases mémoire nécessaires (ici 12 : soit 4 cases pour **x**, 4 pour **y** ou **z** et 4 pour **u**). On travaillera de préférence sur la grammaire initiale.

▷ **Question 4** (2 points)

On modifie la définition des expressions (règles 7 et 8) de la manière suivante :

$$\text{exp} \rightarrow \text{exp} + \text{exp} \mid \mathbf{idf} := \text{exp} \mid (\text{exp}) \mid \mathbf{idf} \mid \mathbf{num}$$

L'affectation devient maintenant une expression, comme dans le langage C. Montrer que cette grammaire est ambiguë. L'opérateur **:=** étant moins prioritaire que l'opérateur **+** et l'opérateur **+** étant associatif à gauche, donner une grammaire non ambiguë respectant cette priorité.

▷ **Question 5** (3 points)

Les programmes d'analyse LL(1) vu en cours s'arrêtent à la première erreur. Cette approche n'est pas réaliste lorsqu'on traite des langages un peu conséquents. L'outil ANTLR propose la stratégie par défaut suivante : si une erreur apparaît dans la procédure A on produit un message d'erreur, on lit jusqu'à trouver un mot dans Suivant(A) (ou la fin de fichier représenté par **\$**) et on stoppe la procédure A.

Reprendre le code de la procédure **bloc** pour planter cette stratégie. On considère les deux erreurs suivantes (exemple de la question 3) :

- Cas a : oubli du ; après le mot réservé **integer**
- Cas b : oubli du **begin** en ligne 3

Expliquer comment se repositionne l'analyseur pour ces deux cas. Cette stratégie est-elle satisfaisante ? Comment pourrait-elle être améliorée ?

Examen Traitement du Signal - ENSIMAG - 1A

19 mai 2009

DOCUMENTS ET CALCULATRICES AUTORISES

1 Propriétés à l'ordre 2 d'un signal déterministe (5 points)

On considère le signal (fonction réelle de la variable réelle)

$$x(t) = \frac{\sin\left[\frac{\pi}{T}t\right]}{\frac{\pi t}{T}} \text{ avec } T > 0, t \in \mathbb{R}$$

1. Montrer que ce signal est d'énergie finie.
2. Calculer sa densité spectrale d'énergie $S_{xx}(\nu)$.
3. Calculer sa fonction d'autocorrélation $\Gamma_{xx}(\tau)$.
4. Quelle est l'énergie de ce signal ?
5. Est-il possible d'échantillonner $x(t)$ en respectant les conditions du théorème d'échantillonnage et si oui, quelle est la fréquence d'échantillonnage minimale ? Même question pour le signal $x(t) = e^{-\pi t^2}$.

2 Modulation (8 points)

On considère un signal analogique réel d'énergie finie $x(t)$ à bande limitée $[-\frac{B}{2}, +\frac{B}{2}]$.
On note $S_{xx}(\nu)$ sa densité spectrale d'énergie.

1. Est-il possible d'échantillonner $x(t)$ en respectant les conditions du théorème d'échantillonnage ? Si oui, quelle est la fréquence d'échantillonnage minimale (pour un échantillonnage régulier) ? Donner une formule de reconstruction du signal analogique à partir des échantillons ?

2. On construit $y(t) = x(t)e^{i2\pi\nu_0 t}$ et $z(t) = \Re[x(t)e^{i2\pi\nu_0 t}]$ (\Re désigne la partie réelle d'un nombre complexe).
 - (a) Exprimer la densité spectrale d'énergie $S_{yy}(\nu)$ du signal modulé $y(t)$.
 - (b) Exprimer la densité spectrale d'énergie $S_{zz}(\nu)$ de la partie réelle $z(t)$ de $y(t)$.
 - (c) Exprimer la fonction de corrélation $\Gamma_{yy}(\tau)$ du signal modulé $y(t)$.
 - (d) Exprimer la fonction de corrélation $\Gamma_{zz}(\tau)$ de $z(t)$.
3. En pratique, on souhaite retrouver $x(t)$ à partir de $z(t)$; quelle solution proposez-vous ?
4. Est-il possible d'échantillonner $z(t)$ sans perte d'information ?

3 Prédition d'un signal aléatoire (7 points)

Dans cet exercice :

- Les signaux sont numériques : ce sont des suites discrètes de valeurs réelles.
- $\{b_k\}_{k \in \mathbb{Z}}$ est un bruit blanc strict réel centré de variance σ^2 (une suite de variables aléatoires centrées indépendantes identiquement distribuées).

Le bruit $\{b_k\}_{k \in \mathbb{Z}}$ entre dans un filtre de transfert $H(z) = [1 - \alpha z^{-1}]$ avec $\alpha \in \mathbb{R}^*$. On note y_k la sortie de ce filtre ($y_k = [H(z)] b_k$).

1. Le filtre $H(z)$ est-il causal ? Est-il stable ? Justifier vos réponses.
2. La sortie y_k du filtre $H(z)$ est-elle centrée ?
3. La sortie y_k du filtre est-elle stationnaire à l'ordre deux ?
4. Exprimer $\mathbb{E}[y_k y_{k-n}]$ la fonction de corrélation de y_k .
5. Calculer le meilleur prédicteur linéaire de y_k en fonction du passé y_{k-1}, y_{k-2} . Le critère choisi pour l'optimisation est l'erreur quadratique moyenne minimale. Quelle est l'erreur de prédiction minimale ?
6. On considère le vecteur d'observation $\mathbf{y}_k = [y_{k-1}, y_{k+1}]^T$. Donner la forme du prédicteur de y_k linéaire, optimal en moyenne quadratique, à partir de l'observation \mathbf{y}_k (une valeur passée et une valeur future) ?
7. Quelle peut être l'utilité d'un tel prédicteur ?
8. Sans faire de calcul, commenter la qualité de cette prédiction par rapport à une prédiction basée sur l'échantillon de même taille $[y_{k-1}, y_{k-2}]$.