EISTI **Année scolaire 2009-2010**

ING1. PÔLE INFORMATIQUE THÉORIQUE

PROJET DE 2ÈME SEMESTRE. CADRE GÉNÉRAL.

THÈME : Codage sans préfixe et automates finis

PROFESSEURS : Hervé de Milleville, Anya Désilles, Yannick Lenir

---

INTRODUCTION

Les projets du pôle "Informatique théorique" ont pour objectif de permettre aux élèves de faire une synthèse des enseignements reçus, en les appliquant à un cas d'étude concret.

Chaque sujet portera sur le codage et décodage de Huffman d'un extrait de texte.

- **Livrable 1. Construction de code de Huffman et encodage des données**
- **Livrable 2. Construction d'automate d'états finis pour le décodage**
- **Livrable 3. Réalisation, en XSL, des traitements de décodage**

MATIÈRES CONCERNÉES.

- XML
- Théorie des langages
- Théorie de l'information

CONNAISSANCES THÉORIQUES VISÉES ET ÉVALUÉES

- Codage de Huffmann
- Graphes, arbres binaires
- Notion de langage,définition d'une grammaire
- Notion d'automate de reconnaissance d'un langage
- Notion d'automate minimal
- Transformation XSL

COMPÉTENCES VISÉES

- savoir concevoir des algorithmes pour les arbres
- savoir construire un code de Huffman pour un texte et un alphabet donnés
- savoir construire l'arbre associé à un code binaire
- savoir reconnaître un code sans préfixe à partir de son arbre
- savoir construire un automate reconnaissant les mots d'un code sans préfixe à partir de son arbre
- savoir construire l'automate minimal pour un automate donné
- savoir mettre en place des traitements XSL à partir de la description XML des données

---

DÉROULEMENT DES PROJETS

Tous les projets se présenteront en quatre parties décrites ci-dessous :

Livrable 1. Construction de code de Huffman L'objectif de cette première partie sera de réaliser un programme en Scilab qui permet de construire un code de compression de Huffman pour un texte donné. Chaque groupe appliquera son programme à un texte qui lui sera fourni. **Ce travail sera en partie réalisé pendant un TP du cours "Théorie de l'information", consacré au codage de Huffman.**
Vous devez livrer :
- Votre programme scilab qui calcule le code de compression de Huffman et encode le texte donné à l'aide de ce code.
- Un fichier texte contenant la suite binaire obtenue par le codage de Huffman
- Un rapport

Le rapport de ce livrable doit présenter en détails :
- La méthode de codage de Huffman
- Les caractéristiques du texte que vous avez traité : entropie, tableau de fréquences des caractères
- Les caractéristiques du codage obtenu : longueur moyenne des mots de code, taux de compression

Livrable 2. Automate de décodage L'objectif de cette étape est de concevoir l'automate d'états finis capable de décoder n'importe quelle séquence obtenue à l'aide du code binaire que vous avez obtenu par la méthode de Huffman. En particulier, cet automate doit permettre de décoder le texte que vous avez compressé. Pour cela, suivez les étapes suivantes :

1. construire l'arbre associé à votre code de Huffman.
2. en reprenant précisément la structure de cet arbre vous pouvez construire un premier automate qui accepte les mots de votre code ; les états finaux de cet automate correspondront aux mots eux mêmes.
3. à partir de ce premier automate construire l'automate minimal acceptant les mots du code
4. réaliser sous JFLAP les modèles des deux automates et les tester

Vous devez livrer :
- Les modèles (fichiers XML) JFLAP des deux automates
- Un rapport

Le rapport de ce livrable doit présenter en détails :
- Les particularité d'un code de Huffman en tant que langage sur alphabet binaire
- La méthode de représentation d'un code binaire par un arbre
- Les particularité de l'arbre associé à un code comme celui de Huffman
- La méthode de l'obtention de l'automate d'états finis acceptant les mots du code et sa justification
- La méthode de construction d'automate minimal ; combien d'états finaux a cet automate ?

Livrable 3. Transformation XSL pour décodage Vous aurez remarqué que sous JFLAP vous ne pouvez testez votre automate que sur des mots séparés qu'il acceptera ou non. Le problème de décodage dans ce projet consiste à parcourir une longue suite **ininterrompue** de bits obtenue par encodage de tout un texte et constituée donc de plusieurs mots du code. Nous avons donc besoin de reconnaître les mots un par un dans l'ordre pour les "traduire" en caractères initiaux et retrouver le texte de départ.

L'objectif de cette étape est d'écrire un programme qui permettrait de décoder une séquence de plusieurs mots à l'aide de l'automate conçu à l'étape précédente. Pour cela vous allez utiliser la structure XML du fichier JFLAP qui représente l'ensemble d'états et de transitions de votre automate et écrire le programme de décodage en XSL.

Voici les indications :

1. Modifiez vos données XML, en y ajoutant l'encodage de la table du code. Cela vous permettra de traduire chaque mot reconnu en caractère de l'alphabet initial. Vous devez proposer une XSD pour le fragment XML que vous allez ajouter.
2. Le template principal de votre XSL doit prendre pour paramètre la chaine de bits qu'il faut décoder.
3. Les templates des éléments "state" et "transition" de JFLAP doivent prendre en paramètre le bit lu et permettre le changement d'état.
4. Tout au long de lecture de bits ils doivent être retirés de la chaîne à décoder et cumulés dans une chaîne de réserve jusqu'à ce qu'un état final soit atteint.
5. le traitement de l'état final doit permettre de traduire le mot de code reconnu et inscrire la traduction dans la sortie et passer l'automate à son état initial en lisant le caractère suivant.

Vous devez livrer :
– la XSD des éléments ajouté dans vos donnés XML
– la feuille de style XSL
– un rapport
Le rapport de ce livrable doit présenter en détails :
– l'algorithme de transformation
– les résultats obtenus
– une conclusion globale du projet et analyse critique des résultats obtenus