

Compresseur de Huffman

Contents

1 Analyse

1.1 Types de données abstraits

Nom: ArbreDeHuffman

Paramètre: Element

Utilise: Naturel, Booleen

Opérations: arbreDeHuffman: $\text{Element} \times \mathbf{Naturel} \rightarrow \text{ArbreDeHuffman}$

combiner: $\text{ArbreDeHuffman} \times \text{ArbreDeHuffman} \rightarrow \text{ArbreDeHuffman}$

obtenirPondération: $\text{ArbreDeHuffman} \rightarrow \mathbf{Naturel}$

obtenirElement: $\text{ArbreDeHuffman} \rightarrow \text{Element}$

estUneFeuille: $\text{ArbreDeHuffman} \rightarrow \mathbf{Booleen}$

obtenirArbreGauche: $\text{ArbreDeHuffman} \rightarrow \text{ArbreDeHuffman}$

obtenirArbreDroit: $\text{ArbreDeHuffman} \rightarrow \text{ArbreDeHuffman}$

Axiomes:

- obtenirElement (arbreDeHuffman(élément,pondération)) = élément
- obtenirPondération (arbreDeHuffman(élément,pondération)) = pondération
- obtenirPondération (combiner(arbreGauche,arbreDroit)) = obtenirPondération(arbreGauche)+obtenirPondération(arbreDroit)
- estUneFeuille (arbreDeHuffman(élément,pondération))
- non estUneFeuille (combiner(arbreGauche,arbreDroit))
- obtenirArbreGauche (combiner(arbreGauche,arbreDroit)) = arbreGauche

Préconditions: obtenirElement(arbre)estUneFeuille(arbre)
non estUneFeuille(arbre)
obtenirArbreGauche(arbre):obtenirArbreDroit(arbre): non estUneFeuille(arbre)

Nom: Octet

Utilise:	Booleen , Bit, Naturel
Opérations:	<p>octet: $\text{Bit} \times \text{Bit} \times \text{Bit} \times \text{Bit} \times \text{Bit} \times \text{Bit} \times \text{Bit} \times \text{Bit} \rightarrow \text{Octet}$</p> <p>estEgal: $\text{Octet} \times \text{Octet} \rightarrow \mathbf{Booleen}$</p> <p>obtenirBit: $\text{Octet} \times 1..8 \rightarrow \text{Bit}$</p>
Sémantiques:	<p>octet: Crée un octet avec les 8 bits spécifiés.</p> <p>estEgal: Vérifie si les deux octets sont égaux.</p> <p>obtenirBit: Obtient le bit à la position spécifiée dans l'octet (position de 0 à 7).</p>
Axiomes:	<ul style="list-style-type: none"> - obtenirBit (00000001, 1) = 1 - estEgal (octet1, octet2) = (obtenirBit (octet1, 1..8) = obtenirBit (octet2, 1..8))
Nom:	Statistiques
Paramètre:	Element
Utilise:	Naturel , Booleen , Ensemble
Opérations:	<p>statistiques: $\text{Ensemble d'Element} \rightarrow \text{Statistiques}$</p> <p>incrementerOccurenceElement: $\text{Statistiques} \times \text{Element} \rightarrow \text{Statistiques}$</p> <p>obtenirOccurenceElement: $\text{Statistiques} \times \text{Element} \rightarrow \mathbf{Naturel}$</p>
Axiomes:	<ul style="list-style-type: none"> - obtenirOccurenceElement(statistiques(set), elem) = 0 - obtenirOccurenceElement(incrementerOccurenceElement(stat, elem) = obtenirOccurenceElement(stat, elem) + 1
Nom:	TableDeCodage
Paramètre:	Element, Code
Utilise:	Naturel , Booleen
Opérations:	<p>TableDeCodage: $\text{TableDeCodage} \rightarrow$</p> <p>estVide: $\text{TableDeCodage} \rightarrow \mathbf{Booleen}$</p> <p>elementPresent: $\text{TableDeCodage} \times \text{Element} \rightarrow \mathbf{Booleen}$</p> <p>codePresent: $\text{TableDeCodage} \times \text{Code} \rightarrow \mathbf{Booleen}$</p> <p>ajouterElement: $\text{TableDeCodage} \times \text{Element} \rightarrow \text{TableDeCodage}$</p> <p>elementPossedeCode: $\text{TableDeCodage} \times \text{Element} \rightarrow \mathbf{Booleen}$</p>

assignerCodeElement: TableDeCodage \times Element \times Code \rightarrow TableDeCodage

obtenirCodeElement: TableDeCodage \times Element \rightarrow Code

obtenirElementCode: TableDeCodage \times Code \rightarrow Element

- Axiomes:**
- non(estVide(ajouterElement(table, elem)))
 - estPresent(ajouterElement(table, elem), elem)
 - elementPresent(ajouterElement(table, elem), elem)
 - codePresent(assignerCodeElement(table, elem, code), code)
 - obtenirElemCode(assignerCodeElement(table, elem, code), code) = elem
 - obtenirCodeElem(assignerCodeElement(table, elem, code), elem) = code
 - non(codePresent(table, code)) \Rightarrow obtenirElementCode(table, code) = NIL

Préconditions: ajouterElement(table, elem) non(elementPresent(table, elem))

elementPresent(table, elem)

elementPossedeCode(table, elem): assignerCodeElement(table, elem, code):
non(codePresent(table, code)) ET elementPresent(table, elem)

obtenirCodeElement(table, elem): elementPossedeCode(table, elem) ET elementPresent(table, elem)

1.2 Analyse descendante