Informatique théorique : feuille numéro 8
Relations d'équivalence

Définitions

Soit R une relation sur l'ensemble E.

1. R est une *relation d'équivalence* si R est réflexive, symétrique et transitive.
2. si R est une relation d'équivalence, la *classe d'équivalence* d'un élément x de E est l'ensemble $\bar{x} = \{y \in E \mid yRx\}$.

Propriétés

les classes d'équivalences de R forment une partition de l'ensemble E :
- $\bar{x} = \bar{y}$ si et seulement si xRy
- $\bar{x} \cap \bar{y} = \emptyset$ si et seulement si $\neg\, xRy$

Définitions

Soit $\sim$ une relation d'équivalence.

1. L'ensemble des classes d'équivalence de $\sim$ est noté $E/\sim$ et appelé *ensemble quotient* de E par la relation :
   $E/\sim = \{\bar{x} \mid x \in E\}$

Exemples

- Sur $E = \{1, 2, 3\}$, soit la relation $R = \{(1,1), (1,2), (2,1), (2,2), (3,3)\}$. On a $\bar{1} = \{1, 2\} = \bar{2}$, $\bar{3} = \{3\}$, $E = \bar{1} \cup \bar{3}$, $E/R = \{\bar{1}, \bar{3}\}$.
- la relation de congruence modulo $n \in \mathbb{N} \setminus \{0, 1\}$ sur $\mathbb{Z}$
- L'accessibilité dans un graphe non orienté.
  Quelles sont les classes d'équivalence ?
- La relation "être composée des mêmes lettres" sur les chaînes de caractères
- La relation "avoir la même longueur" sur les listes
- La relation "avoir la même taille" sur des fichiers
- La relation "avoir la même valeur de hachage" sur des clés.
  Que peut-on dire des classes d'équivalence lorsque la fonction de hachage est uniforme (chaque clé a autant de chances d'être hachée vers l'une quelconque des valeurs de hachage) ?

1 Exercice

- Soit R une relation de pré-ordre (reflexive et transitive) sur l'ensemble E. Montrer que $R \cap R^{-1}$ est une relation d'équivalence.

– Soit R une relation sur l'ensemble E. La relation $(R \cup R^{-1})^\star$ est-elle une relation d'équivalence ?

2 Exercice

On considère la relation R définie sur $\mathbb{R}^2$ par la condition suivante : $(x, y)R(x', y')$ si, et seulement si, $x = x'$.

1. Montrer que R est une relation d'équivalence.
2. Quelles sont les classes d'équivalence ?

3 Exercice

Soit E et F deux ensembles et f une fonction de E dans F. Soit R la relation definie sur E par : xRy si et seulement si $f(x) = f(y)$.

1. Montrer que R est une relation d'equivalence sur E.
2. Demontrer que si f n'est pas injective, il existe au moins une classe qui contient deux éléments ou plus.
3. Donner des exemples de fonction sur les tableaux et de classes d'équivalence associées.

4 Exercice

Soit $\sim$ la relation définie sur les propositions logiques utilisant quatre variables `P`, `Q`, `R` et `S`, par :
`A` $\sim$ `B` si et seulement si on a l'implication sémantique $\models$ `A` $\Leftrightarrow$ `B`.

1. Montrer que chaque classe d'équivalence est infinie.
2. Combien y a-t-il de classes d'équivalence pour cette relation ?
3. Si `A` $\sim$ `B` et `C` $\sim$ `D`, que peut-on dire de `A` $\wedge$ `C` et `B` $\wedge$ `D`,
4. Si `A` $\sim$ `B` et `C` $\sim$ `D`, que peut-on dire de `A` $\vee$ `C` et `B` $\vee$ `D` ?

5 Exemple de l'API Java

public boolean *equals*(Object obj)
The equals method implements an equivalence relation on non-null object references
The equals method for class *Object* implements the most discriminating possible equivalence relation on objects ; that is, for any non-null reference values x and y, this method returns *true* if and only if x and y refer to the same *object* ($x == y$ has the value *true*).

public interface $Comparable < T >$.
This interface imposes a total ordering on the objects of each class that implements it. This ordering is referred to as the class's natural ordering, and the class's *compareTo* method is referred to as its natural comparison method.

The natural ordering for a class C is said to be consistent with *equals* if and only if $e1.compareTo(e2) == 0$ has the same boolean value as $e1.equals(e2)$ for every $e1$ and $e2$ of class C.
For the mathematically inclined, the relation that defines the natural ordering on a given class C is :

$$\{(x, y) \text{ such that } x.compareTo(y) <= 0\}.$$

The quotient for this total order is :

$$\{(x, y) \text{ such that } x.compareTo(y) == 0\}.$$

It follows immediately from the contract for *compareTo* that the quotient is an equivalence relation on C, and that the natural ordering is a total order on C. When we say that a class's natural ordering is consistent with *equals*, we mean that the quotient for the natural ordering is the equivalence relation defined by the class's $equals(Object)$ methode.

$$\{(x, y) \text{ such that } x.equals(y)\}.$$

5.1 questions

- Si vous redéfinissez la méthode *equals* dans une sous-classe de la classe *Object*, que peut-on dire de la nouvelle relation d'équivalence ainsi définie.
- Dans l'interface *Comparable* il est question de quotient par rapport à une relation d'ordre. De quel quotient s'agit-il en fait ?