NGUYEN Thi Minh Tuyen

Théorie des automates et langages formels

Expressions régulières

Une nouvelle méthode pour définir les langages

- **alphabet → langage**
  - $S = \{x\}$ $S^* = \{\Lambda, x, xx, xxx, \ldots\}$

    ou simplement $\{x\}^* = \{\Lambda, x, xx, xxx, \ldots\}$
- **langage → langage**
  - $S = \{xx, xxx\}$ $S^* = \{\Lambda, xx, xxx, xxxx, \ldots\}$

    ou simplement $\{xx, xxx\}^* = \{\Lambda, xx, xxx, xxxx, \ldots\}$
- **« lettre » → langage**
  - $\mathbf{x^*}$ (écrit en caractères gras)
  - $\text{langage}(\mathbf{x^*}) = \{\Lambda, x, xx, xxx, \ldots\}$

  ou simplement $\mathbf{x^*} = \{\Lambda, x, xx, xxx, \ldots\}$

Une nouvelle méthode pour définir les langages

- langage(ab^*) = {a, ab, abb, abbb, …}
- langage($(ab)^*$) = {Λ, ab, abab, ababab, …}

Plusieurs façons d'exprimer le même langage

– {x, xx, xxx, xxxx, …}

– xx^* x^+ xx^*x^* x^*xx^* $(x^+)x^*$ $x^*(x^+)$ $x^*x^*xx^*$

- langage(a^*b^*) =

{Λ, a, b, aa, ab, bb, aaa, aab, abb, bbb, aaaa, …}

(les « a » avant les « b »)

- <u>Remarque</u>: langage(a^*b^*) ≠ langage($(ab)^*$)

Exemple: S-IMPAIR ($L = \{x^{impair}\}$)

- **Règle 1**: x ∈S-IMPAIR
- **Règle 2**: Si w appartient à S-IMPAIR alors xxw appartient à S-IMPAIR

- S-IMPAIR = langage(**x**(**xx**)*)
- S-IMPAIR = langage((**xx**)*__x__)
- Par contre S-IMPAIR ≠ langage(**x*xx***)

xx|x|x

Symbole +

- Le symbole + est utile pour décrire les langages.
- x + y : on peut choisir x ou bien y dans le mot.
- Exemple:
  - $\Sigma = \{a, b, c\}$
  - T = {a, c, ab, cb, abb, cbb, abbb, cbbb, ...}
    = langage((a+c)b*)

Exemple

- L = {aaa, aab, aba, abb, baa, bab, bba, bbb}

  Langage des mots de longueur 3 sur l'alphabet {a, b}
- L = langage((a+b)(a+b)(a+b))

  = langage($(a+b)^3$)
- **(a+b)*** décrit le langage de tous les mots sur l'alphabet {a,b}
- **a(a+b)*** = ?
- **a(a+b)*b** = ?

Définition formelle des expressions régulières

L'ensemble des expressions régulières sur un alphabet Σ est défini récursivement comme suit:

1. Chaque lettre de Σ ainsi que Λ est une expression régulière.
2. Si $\mathbf{r_1}$ et $\mathbf{r_2}$ sont des expressions régulières , alors il en est de même pour:
    1. (r_1)
    2. $r_1 r_2$
    3. $r_1 + r_2$
    4. $r_1{}^*$
3. Aucune autre expression n’est régulière.

Définition formelle des expressions régulières

- Noter que $\mathbf{r_1^+ = r_1 r_1^*}$
- $\mathbf{r_1 = r_2}$ si et seulement si langage($\mathbf{r_1}$) = langage($\mathbf{r_2}$)
- Exemple: **(a+b)*a(a+b)***

  Les mots avec au moins un « a »

  abbaab: (Λ)a(bbaab) (abb)a(ab) (abba)a(b)
- Les mots sans aucun « a »?

  b*
- Tous les mots de {a,b}?

  (a+b)*a(a+b)* + b*

  Donc: **(a+b)* = (a+b)*a(a+b)* + b***

Exemples

- **Exemple 1:**
  - **Le langage de tous les mots avec aux moins deux « a ».**

$$(a+b)^*a(a+b)^*a(a+b)^*$$
$$= b^*ab^*a(a+b)^*$$
$$= (a+b)^*ab^*ab^*$$
$$= b^*a(a+b)^*ab^*$$

- **Exemple 2:**
  - **Le langage de tous les mots avec exactement deux « a ».**

$$b^*ab^*ab^*$$

Exemples

Tous les mots avec au moins un « a » et au moins un « b ».

Solution 1:

(a+b)*a(a+b)*b(a+b)* + (a+b)*b(a+b)*a(a+b)*

- Mais **(a+b)*a(a+b)*b(a+b)*** exprime tous les mots sauf ceux qui sont formés d'une suite de « b » (au moins un) suivie d'une suite de « a » (au moins un).

bb*aa*

Solution 2:

(a+b)*a(a+b)*b(a+b)* + bb*aa*

- Donc: **(a+b)*a(a+b)*b(a+b)* + (a+b)*b(a+b)*a(a+b)*** = **(a+b)*a(a+b)*b(a+b)* + bb*aa***

Exemples

- Les seuls mots qui ne contiennent pas en même temps des « a » et des « b »:

$$a^* + b^*$$

- Donc:

$$(a+b)^* = (a+b)^*a(a+b)^*b(a+b)^* + bb^*aa^* + a^* + b^*$$

Concaténation

- Exemple: Le langage des mots sans « a » ainsi que les mots qui commencent avec « a » suivi de quelques b (peut être 0 « b »):

$$\{\Lambda, a, b, ab, bb, abb, bbb, abbb, bbbb, \ldots\}$$

$$b^* + ab^* \qquad (\Lambda + a)b^*$$

- En générale: concaténation est distributif par rapport à l'opération +.

$$r_1(r_2 + r_3) = r_1r_2 + r_1r_3$$

$$(r_1 + r_2)\, r_3 = r_1r_3 + r_2r_3$$

Exemple de la règle de distributivité

$(a+c)b^* = ab^*+cb^*$

- 2 opérations: langage(s) → langage
  - Si S et T sont deux langages sur un même alphabet Σ,
    1. S+T: la réunion des langages S et T défini comme $S \cup T$
    2. ST : le produit de concaténation défini comme l'ensemble des mots x qui s'écrivent sous la forme vw avec v est un élément de S et w est un élément de T.
- Exemple: S = {a, bb} T = {a, ab}

ST = {aa, aab, bba, bbab}

Langage associé à une expression régulière

Défini par les règles récursives suivantes:

1. Le langage associé à une seule lettre est le langage qui contient un seul mot formé par cette lettre. Le langage associé à **Λ** est {Λ}.
2. Si L_1 est le langage associé à l'expression régulière r_1 et L_2 est le langage associé à l'expression régulière r_2:
   i. Le produit de concaténation L_1L_2 est le langage associé à l'expression régulière r_1r_2: langage$(r_1r_2) = L_1L_2$
   ii. La réunion L_1+L_2 est le langage associé à l'expression régulière r_1+r_2: langage$(r_1 +r_2) = L_1+L_2$
   iii. L'étoile de L_1, L_1^*, est le langage associé à l'expression régulière r_1^*: langage$(r_1^*) = L_1^*$

Remarque: Toute expression régulière à un langage qui lui est associé.

Les langages finis sont réguliers

- **Théorème:**

  Soit L un langage fini, il existe une expression régulière qui définit L.
- **Algorithme (et Preuve)**

  Écrire chaque mot de L en caractères gras, et écrire un « **+** » entre ces mots: ceci nous donne l'expression régulière correspondante.
- **Exemple:** L = {baa, abbba, bababa}

  baa + abbba + bababa
- l'expression régulière définie par cet algorithme n'est pas unique:
- **Exemple:** L = {aa, ab, ba, bb}

  aa + ab + ba + bb ou **(a+b)(a+b)**
- <u>Remarque:</u> Cet algorithme ne fonctionne pas pour les langages infinis.

Exemples

- L = {Λ, x, xx, xxx, xxxx, xxxxx}
- L'expression régulière obtenue de l'algorithme est

  Λ + x + xx + xxx + xxxx + xxxxx

  ou

  (Λ + x)^5

Exemples

- L'opération étoile appliquée à une sous-expression qui contient une étoile

  (a+b*)* **(aa+ab*)***

  (a+b*)* = (a+b)* **(aa+ab*)* ≠ (aa+ab)*** abb|abb

- **(a*b*)***
  - Les lettres « a » et « b » appartiennent au langage **(a*b*)**.
  - Donc **(a*b*)* = (a+b)***
- Est-il possible de déterminer si deux expressions régulières sont équivalentes?
  - Avec un ensemble de règles algébriques?
    - On ne le sait pas.
  - Avec un algorithme?
    - Oui.

Exemples

- Les mots avec double lettre:

$$(a+b)^*(aa+bb)(a+b)^*$$

- Sans double lettre:

$$(\Lambda+b)(ab)^*(\Lambda+a)$$

Donc:

$$(a+b)^* = (a+b)^*(aa+bb)(a+b)^* + (\Lambda+b)(ab)^*(\Lambda+a)$$

Exemples

- Langage PAIR-PAIR défini par l'expression

  [aa + bb + (ab + ba)(aa+bb)*(ab + ba)]*

- Chaque mot de PAIR-PAIR contient un nombre pair des « a » et des « b ».
- Chaque mot qui contient un nombre pair des « a » et des « b » est un élément de PAIR-PAIR.

Question?