Approche Algébrique de la Représentation et du Raisonnement Temporels : S-langages

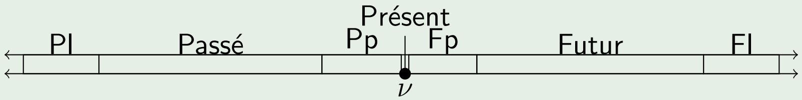
Sylviane R. Schwer

LIPN, UMR 7030, CNRS & Université Paris 13

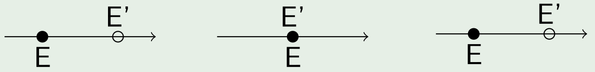
Séminaire MaMuX, IRCAM 12/03/2010

Conceptions du temps : séries A, B, C de McTaggart (1908)

Série A : déictique (fleuviste d'Héraclite, prédicative)



Série B : relationnelle (éternaliste de Parménide)



E' antérieur E   E' simultané E   E' postérieur E

Série C : chronologique (structure ordinale)

U	-101	-100	-99	···	-80	···	-45	-44	-43
		E'						E	

S-langages : thèses

- On ne représente pas le temps mais des relations temporelles.
- Seul ce qui est saillant est pertinent.
- Les relata sont in fine des points soit comme objets de l'ontologie soit comme des bornes associées à des *changements détat* de ces objets.
- Représenter et raisonner sur les relations temporelles qualitatives relèvent du calcul algébrique, non de la logique : c'est une analyse situs entre séries de bornes ponctuelles.
- Deux relations à représenter: précédence et simultanéité

Associer une identité à chacune des entités.

- Les bornes (points isolés ou bornes d'intervalles) sont représentées par l'identité de l'entité correspondante,
- des étiquettes (indices, diacritiques) permettent d'encoder des informations spécifiques, comme des dates, ou des informations sur le type de bornes (objets lui-même, début /fin d'un état, etc.)
- On travaille sur un langage de bornes, manipulés comme des points

S-langages : justification

- La théorie des langages formels classiques :

précédence : Oui
Correspondance entre l'ordre de lecture et l'ordre des occurrences dans le temps.

ab signifie que l'objet d'identité *a* précède celui d'identité *b* (ou que l'objet d'identité *b* succède celui d'identité *a*)

simultanéité : Non
Les langages de traces $[ab \equiv ba]$ sont insuffisants car ne permettent pas de distinguer concurrence et simultanéité.

- Théorie des langages formels sur des paquets de lettres avec possibilité d'accéder aux lettres.

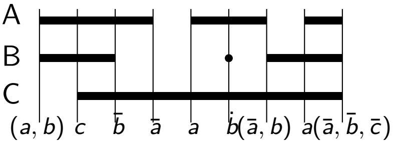
S-langages : Les objets

théorie des langages formels sur des paquets de lettres avec possibilité d'accéder aux lettres.

Définitions

- un n-alphabet de base $\mathcal{A} = \{a_1, \cdots, a_n\}$.
- le S-alphabet associé $2^{\mathcal{A}} - \emptyset$.
- une S-lettre : un élément du S-alphabet.
  C'est une partie non vide du S-alphabet.
- un S-mot : une suite de S-lettres.
- un S-langage : ensemble de S-mots.
- Vecteur de Parikh d'un S-mot u: $(\#\mathcal{A})$-uplet $\vec{u}_{\mathcal{A}} = (u_1, \cdots, u_n)$ qui associe à chaque lettre a_i son nombre d'occurences u_i dans le S-mot.
- S-langage Univers d'un vecteur de Parikh $(p_1, \cdots, p_n)$ est l'ensemble $\mathcal{U}(p_1, \cdots, p_n)$ des S-mots u ayant ce vecteur comme vecteur de Parikh, *i.e.* tels que $\vec{u}_{\mathcal{A}} = (p_1, \cdots, p_n)$.

S-langages
Exemple



son S-mot $(a\bar{a})^3$

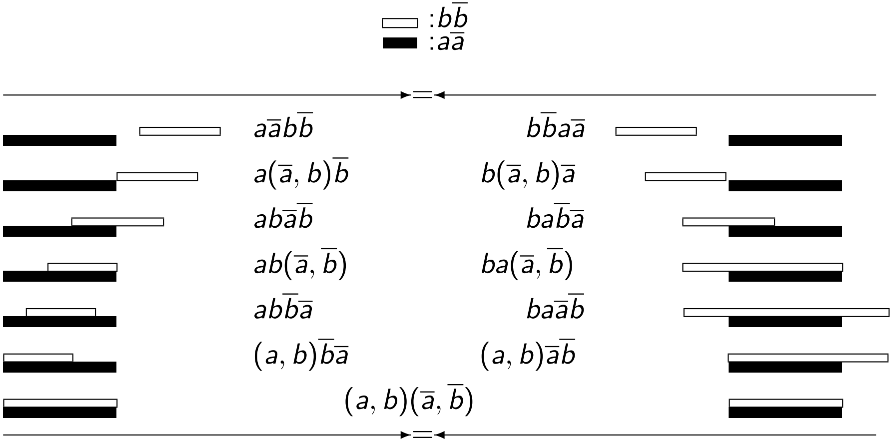
son S-mot $b\bar{b}\dot{b}b\bar{b}$

son S-mot $c\bar{c}$

Le S-mot encodant la relation

- les entités : A, B, C
- l'alphabet : $\mathcal{A} = \{a, b, c\}$, le vecteur de Parikh : (6,5,2)
- le S-alphabet :
  $\widehat{\mathcal{A}} = \{(a), (b), (c), (a, b), (b, c), (a, c), (a, b, c)\}$
- description des objets : $(a)^6$, $(b)^5$,$(c)(c)$
- description de la relation: (a,b)(c)(b)(a)(a)(b)(a,b)(a)(a,b,c)
- Le S-univers :
  $\mathcal{L}(6, 5, 2) = \{u \in \widehat{\mathcal{A}}^* || u|_a = 6, |u|_b = 5, |u|_c = 2\}$

Situations respectives de deux intervalles sur une droite.



S-langages : les opérateurs

- Les opérateurs de la théorie des langages formels : les S-lettres sont des boîtes closes, traitées comme des lettres
  - union,
  - intersection,
  - différence,
  - projection,
  - produit de concaténation,
  - étoile,
  - morphisme,
  - morphisme inverse,
  - shuffle,
  - ...
- Les opérateurs propres : les S-lettres sont des boîtes ouvrables
  - S-projection
  - Produit cartésien $\otimes$
  - Jointure J ou $\bowtie$

S-projection

Soit un alphabet $\mathcal{A}$, un sous-alphabet $\mathcal{B}$ de $\mathcal{A}$ et un S-mot u écrit sur $\mathcal{A}$. La S-projection de u sur $\mathcal{B}$, notée $u_{|\mathcal{A}}$ consiste à

1. effacer dans les S-lettres de u toutes les occurrences de lettres de u qui ne sont pas dans $\mathcal{B}$
2. effacer de u toutes les S-lettres qui sont devenues vides

S-projection : exemple

$\mathcal{A} = \{a, b, c, d\}$
$u = (a, b)(c)(b)(a)(a)(b)(a, b)(a)(a, b, c)$
$u_{|_{\{a,b\}}} = (a, b)(b)(a)(a)(b)(a, b)(a)(a, b)$
$u_{|_{\{c,b,d\}}} = (b)(c)(b)(b)(b)(b, c)$
$uf_{|_{\{c\}}} = (c)(c)$. $uf_{|_{\{c,d\}}} = (c)(c)$.
$uf_{|_{\{d\}}} = \epsilon$, le mot vide.

Jointure de deux S-mots

Soient deux alphabets $\mathcal{A}$ et $\mathcal{B}$ un S-mot u écrit sur $\mathcal{A}$ de vecteur de Parikh $\vec{u}$ et un S-mot v écrit sur $\mathcal{B}$, de vecteur de Parikh $\vec{v}$.
La Jointure de u et de v est le S-langage $u \bowtie v$ sur $\mathcal{A} \cup \mathcal{B}$, ensemble des S-mots de vecteur de Parikh $Max(\vec{u}_{\mathcal{A}\cup\mathcal{B}}, \vec{v}_{\mathcal{A}\cup\mathcal{B}})$ tel que leur S-projection sur $\mathcal{A}$ vaut u et la S-projection sur $\mathcal{B}$ vaut v.

$\Rightarrow u \bowtie v = v \bowtie u$

$\Rightarrow u \bowtie \varepsilon = u$

$\Rightarrow u_{|\mathcal{A}\cap\mathcal{B}} = v_{|\mathcal{A}\cap\mathcal{B}}$

$\Rightarrow$ Si $\mathcal{B} \subseteq \mathcal{A}$, $u \bowtie v = v$

$u = ab(a,b)$; $\qquad v = x(a,x)x(a,x)$
$u \bowtie v = \{x(a,x)bx(a,b,x),$
$x(a,x)(b,x)(a,b,x),\ x(a,x)xb(a,b,x)\}$

$u = ab$; $v = ba$
$u \bowtie v = \emptyset$

S-langage : produit cartésien de deux S-mots

Produit cartésien : définition

Quand les alphabets sont disjoints, on note la S-jointure $\otimes$.

produit cartésien : exemple

$u = aa \qquad v = bb$

$u \otimes v = \{aabb, a(a, b)b, abab, ab(a, b), abba, (a, b)ab, (a, b)(a, b),$
$(a, b)ba, baab, ba(a, b), baba, b(a, b)a, bbaa\}$

C'est l'ensemble associé au positionnement relatif de deux intervalles sur une droite.

produit cartésien : exemple

$u = ac \qquad v = bd$

$u \otimes v = \{acbd, a(c, b)d, abcd, ab(c, d), abdb, (a, b)cd, (a, b)(c, d),$
$(a, b)dc, bacd, ba(c, d), badc, b(a, d)c, bdac\}$

Ensemble isomorphe au précédent.

Méthode de calcul de la S-jointure de deux S-mots.

Soient deux alphabets $\mathcal{A}$ et $\mathcal{B}$ un S-mot u écrit sur $\mathcal{A}$ et un S-mot v écrit sur $\mathcal{B}$

1. On colorie l'alphabet $\mathcal{A} \cap \mathcal{B}$,
2. On colorie toute S-lettre comportant une lettre de$\mathcal{A} \cap \mathcal{B}$ dans la couleur de $\mathcal{A} \cap \mathcal{B}$ dans u et v. On obtient un sous-mot maximal u' de u et un sous-mot maximal v' de v de même couleur que $\mathcal{A} \cap \mathcal{B}$ et qui ont même longueur.
3. On aligne S-lettre par S-lettre u' et v' dans u et v.
4. On fusionne les S-lettres alignées
5. Avant la première S-lettre, après la dernière S-lettre et entre deux S-lettres ainsi construites, se trouve un S-mot de $\mathcal{A} - \mathcal{B}$ ou le S-mot vide et un S-mot de $\mathcal{B} - \mathcal{A}$ ou le S-mot vide dont on fait le produit cartésien.

Méthode de calcul de la S-jointure de deux S-mots.

$\mathcal{A} = \{a, b, c, d\}$, $\mathcal{B} = \{b, d, x, y, z\}$

1. $\mathcal{A} \cap \mathcal{B} = \{b, d\}$
2. $u = a(b, z)czza(b, d, a)zcdc$
   $v = xbyx(b, d, x)dyx$
3.

u	a	(b, z)	c	z	z	a	(b, d, a)	z	c	d	c	
v	x	b	y	x			(b, d, x)			d	y	x

4.

$- \otimes -$	(b, z)	$- \otimes -$	(b, d, a, x)	$- \otimes -$	d	$- \otimes -$

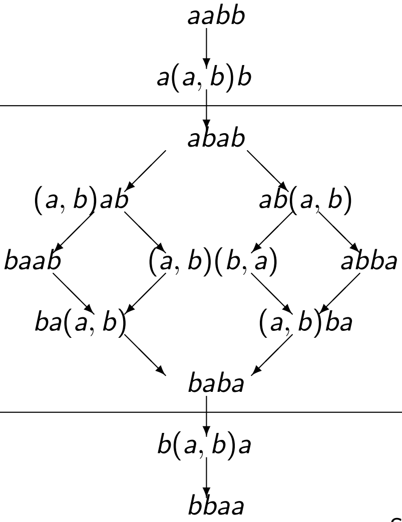
5. $u \bowtie v = [a \otimes x](b, z)[czza \otimes yx](b, d, a, x)zcd[c \otimes yx]$

opérations sur les S-langages

- La S-projection d'un S-langage est le S-langage des S-projections de ses S-mots : $L_{|\mathcal{A}} = \{u_{|\mathcal{A}}\ u \in L\}$
- La jointure de deux S-langages est le S-langage formé des jointures d''un S-mot de l'un avec un S-mot de l'autre $L \bowtie L' = \cup_{u \in L, v \in L'} u \bowtie v$
  - $L \bowtie L' = L' \bowtie L$
  - $(L \bowtie L') \bowtie L'' = L \bowtie (L' \bowtie L'')$
- Soit $\mathcal{A} = \{a_1, \ldots, a_n\}$ et $(p_1, \ldots, p_n)$ un n-uplet d'entiers,
  - $\mathcal{U}(p_1, \cdots, p_n) = (a_1)^{p_1} \otimes \cdots \otimes (a_n)^{p_n}$
  - $\mathcal{U}(p_1, \cdots, p_n)$ possède $D(p_1, \cdots, p_n)$ S-mots, nombre de Delannoy généralisé [Schwer2002].
  - $\mathcal{U}(p_1, \cdots, p_n)$ peut être organisé en treillis généré par le S-mot $a_1^{p_1} \cdots a_n^{p_n}$ et un système de réécriture.

le treillis $\mathcal{U}(2, 2)$

$A < B$ $\qquad$ $a[ab, (a, b)]b$



$A\mathbf{o}B$ $\qquad$ $(a \otimes b)^2$

$A > B$ $\qquad$ $b[ba, (a, b)]a$

Wiener-Kamp relations

S-langages

S-langages élémentaires

Un S-langage est dit élémentaire s'il possède une expression qui n'utilise que des produits (concaténation, étoile et cartésien) sur des S-mots.

- $\mathcal{U}(p_1, \cdots, p_n)$ est élémentaire
- $\mathcal{A}^*$ est élémentaire
- La relation de Wiener-Kamp AoB correspond à un S-langage élémentaire
- La relation de Wiener-Kamp $<$ correspond à un S-langage non élémentaire.
- $\{aabb, bbaa\}$ n'est pas un S-langage élémentaire.
- Dans $\mathcal{U}(2, 2)$, tout S-langage s'exprime comme somme disjointe d'au plus sept S-langages élémentaires.
  Le plus petit tel S-langage est
  $\{aabb, abab, baab, (ab)(ab), abba, baba, bbaa\}$.

décompositions de S-langages

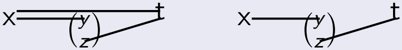
- Le calcul de la jointure est NP-complet.
- Il est facile (complexité polynômiale) de calculer la jointure de S-langages élémentaires.
- Tous les sous-treillis de $\mathcal{U}(p_1, \cdots, p_n)$ ne correspondent pas à des S-langages élémentaires mais ils correspondent au diagramme de Hasse d'un pré-ordre : notion de relation convexe et S-langage convexe.
- Tout S-langage est l'union disjointe de S-langages convexes
- Tout S-langage convexe est l'union disjointe de S-langages élémentaires mais ...
- ... le passage par les diagrammes de Hasse peut être plus simple

pré-ordre partiel et diagramme de Hasse d'un S-univers

Soit $\mathcal{A} = \{a_1, \cdots, a_n\}$ un n-alphabet et $(p_1, \cdots, p_n)$ un n-uplet d'entiers. Un pré-ordre du S-univers $\mathcal{U}(p_1, \cdots, p_n)$ est une structure $< E, r, s >$ telle que

- E est partitionné en n classes $E_1, \cdots, E_n$,
- chaque E_i étant une chaîne de p_i éléments $a_{i,1} - a_{i,2} - \ldots - a_{i,p_i}$
- r est irréflexive, asymétrique et transitive
- s est symétrique et transitive
- r et s préservent les chaînes
- si $x\ r\ y$ et $y\ s\ z$ et $z\ r\ t$, alors $x\ r\ t$

Le diagramme de Hasse du pré-ordre est le graphe minimal associé au pré-ordre



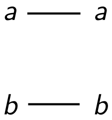
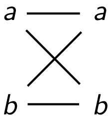
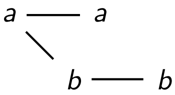
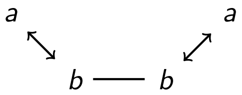
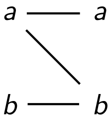
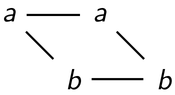
pré-ordre diagramme de Hasse

pré-ordre partiel et S-langage d'un S-univers

Soit $\mathcal{A} = \{a_1, \cdots, a_n\}$ un n-alphabet et $(p_1, \cdots, p_n)$ un n-uplet d'entiers, $\mathcal{U}(p_1, \cdots, p_n)$ le S-univers associé.

- Les classes E_i correspondent à l'énumération ordinale du S-mot $(a_i)^{p_i}$
- Il y a correspondance biunivoque entre un pré-ordre total (ou linéaire) et $\mathcal{U}(p_1, \cdots, p_n)$
- Il y a correspondance biunivoque entre l'ensemble des S-langages convexes et les préordres de $\mathcal{U}(p_1, \cdots, p_n)$.

Quelques S-langages convexes dans $\mathcal{U}(2,2)$ (parmi les 81).

 $aa \otimes bb$ no constraints (13 positions)	 $[a \otimes b][a \otimes b]$ $A \cap B \neq \emptyset$ (9 positions)	 $a[a \otimes bb]$ A begins before B (5 positions)
 $a\ \overset{\overset{1}{\leftrightarrow}}{bb}\ a$ B is a proper part of A (3 positions)	 $[aa \otimes b]b \cup$ $[a \otimes b][(a,b);ba]$ A begins before b ends (11 positions)	 $a[a \otimes b]b$ A begins and ends before B (3 positions)

Raisonnement temporel

Représenter une situation temporelle : placer des événements ou des traces temporelles d'entités dans le temps à partir d'un ensemble d'informations fournies

Raisonner sur une situation temporelle : calculer ou inférer de de nouvelles informations, déterminer si une assertion est compatible ou non avec les informations fournies

Approche qualitative : Seules les formes et les positions relatives (topologiques) entre en compte

Approche quantitative : Informations numériques, les mesures, distances

S-langage

- Dans un premier temps, approche qualitative
- Mais toute mesure étant *in fine* relationnelle, en particulier les mesures calendaires, intégration de l'aspect quantitatif : prise en compte nécessaire de certaines propriétés du domaine numérique
  - domaine continu : une fonction de datation (horo-calendaire) de l'ensemble des bornes dans le domaine continu ou ajout d'un alphabet supplémentaire associer à un S-langage particulier simulant "les top d'horloge"
    hier, j'ai
  - domaine discret : réinterprétation de la relation de simultanéité et de contiguïté

Raisonnement temporel et S-langage

- Toute contrainte temporelle qualitative (conjonctive et disjonctive) s'exprime sous la forme d'un S-langage
- L'ensemble des contraintes temporelles qualitative est la jointure d'un ensemble de S-langages : chaque S-mot correspond à une solution possible.
- Un problème de contraintes temporelles qualitatives est cohérent si et seulement si le S-langage résultant est non vide.

Raisonnement temporel et S-langage

Avantage des S-langages sur les approches d'Allen, Vilain, Ladkin, Ligozat, ...

- Aucune contrainte sur la forme des entités et traitement homogène des entités et des relations
- Pas de tables de calcul : possibilité de traiter des relations n-aires.
- puissance des langages formels : traitement de la granularité par substitution
- la minimalité du système permet de modifier la sémantique pour repésenter tous les systèmes de pré-ordres les hiérarchies, les systèmes calendaires, ...
- intégration possible de séries infinies potentielles
- traitement des séries temporelles

Exemple de raisonnement sur 6 intervalles

Description du problème.

Soit six trains A, B, C, D, E et F et les quatre contraintes temporelles

1. A, B et E arrivent à quai en même temps, mais A repart avant B.
2. A part après ou en même temps que C, mais avant l'arrivée de D.
3. D et F arrivent à l'instant où B part.
4. E et D repartent en même temps.

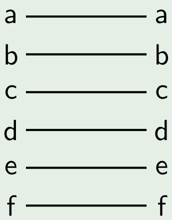
Combien de voies faut-il au minimum pour satisfaire les contraintes ?

Exemple.

Modélisation

- Alphabet : $\mathcal{A} = \{a, b, c, d, e, f\}$
- objets et repésentation : L'intervalle de temps de stationnement en gare pour chaque train.
- Soit le S-langage Univers $\mathcal{U} = aa \otimes bb \otimes cc \otimes dd \otimes ee \otimes ff$

et son diagramme de Hasse :



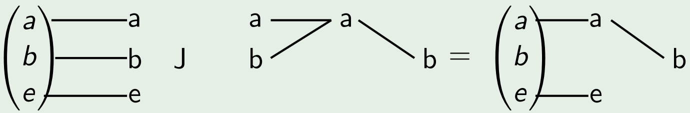
Exemple.
Calcul incrémental

Contraintes 1 : S-langage

1. A, B et E arrivent en même temps : $(a, b, e)[a \otimes b \otimes e]$
2. A repart avant B : $[a \otimes b]ab$

$\mathcal{C}_1 = (a, b, e)[a \otimes b \otimes e] J [a \otimes b]ab = (a, b, e)[ab \otimes e]$

Contraintes 1 : diagrammes de Hasse

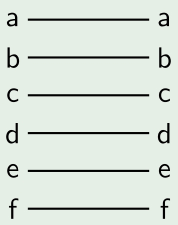


Exemple : calcul incrémental
Satisfaction de C1

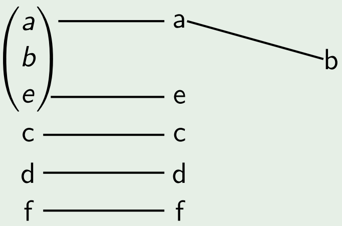
S-langage résultant

Satisfaction de C1 : $\mathcal{U}$ J $(a, b, e)[ab \otimes e]$ soit
$\mathcal{L}_1 = (a, b, e)[ab \otimes e] \otimes cc \otimes dd \otimes ff$

Etat Initial



Etat C1



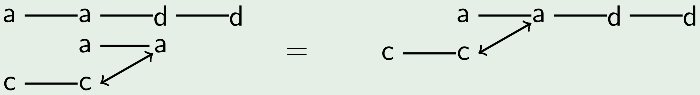
Exemple. Contrainte 2

Contraintes 2 : S-langage

1. A part après ou en même temps que C:
   $[a \otimes cc]a \cup [a \otimes c](a, c) = [a \otimes c\vec{c}]a$
2. A repart avant l'arrivée de D : *aadd*

$\mathcal{C}_2 = [a \otimes c\vec{c}]a$ J $aadd$=$[a \otimes c\vec{c}]add$

Contraintes 2 : diagrammes de Hasse



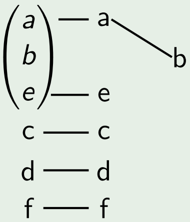
S-langage résultant

$\mathcal{L}_2 = \mathcal{L}_1$ J $\mathcal{C}_2$ soit $(a, b, e)[ab \otimes e] \otimes cc \otimes dd \otimes ff$ J $[a \otimes c\vec{c}]add$
$\mathcal{L}_2 =[[c\vec{c} \otimes (a, b, e)]a[b \otimes dd] \otimes ff]$ J $(a, b, e)e$

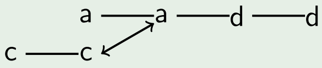
Exemple.

Satisfaction de C1 & C2

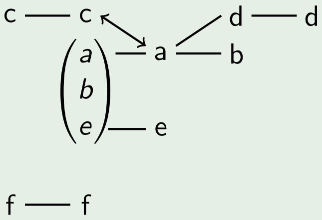
Etat C1



Contrainte C2



Etat C1 & C2



S-langage résultant

$\mathcal{L}_2 = \mathcal{L}_1 \ J \ \mathcal{C}_2$ soit $(a, b, e)[ab \otimes e] \otimes cc \otimes dd \otimes ff \ J \ [a \otimes c\vec{c}]add$

$\mathcal{L}_2 = [[c\vec{c} \otimes (a, b, e)]a[b \otimes dd] \otimes ff] \ J \ (a, b, e)e$

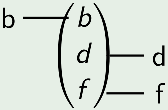
Exemple. Contrainte 3

Contrainte 3 : S-langage

1. D et F arrivent à l'instant où B repart : $[b(b,d,f)[d \otimes f]$

$\mathcal{C}_3 = [b(b,d,f)[d \otimes f]$

Contrainte 3 : diagramme de Hasse



S-langage résultant

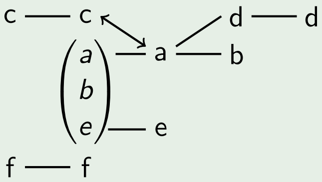
$\mathcal{L}_3 = \mathcal{L}_2 \; \mathrm{J} \; \mathcal{C}_3 = [[c\vec{c} \otimes (a,b,e)]a[b \otimes dd] \otimes ff] \; \mathrm{J} \; (a,b,e)e \; \mathrm{J}$
$[b(b,d,f)[d \otimes f]$

$\mathcal{L}_3 = [c\vec{c} \otimes (a,b,e)]a(d,b,f)[f \otimes d] \; \mathrm{J} \; (a,b,e)e$

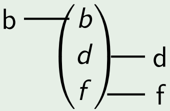
Exemple. Satisfaction de C1 & C2 & C3

$\mathcal{L}_3 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[f \otimes d]\ \mathbf{J}\ (a, b, e)e$

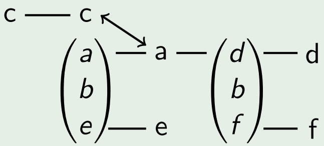
Etat C1 & C2



Contraintes 3



Etat C1 & C2 & C3



S-langage résultant

$\mathcal{L}_3 = \mathcal{L}_2\ \text{J}\ \mathcal{C}_3 = [[c\vec{c} \otimes (a, b, e)]a[b \otimes dd] \otimes ff]\ \text{J}\ (a, b, e)e\ \text{J}$
$[b(b, d, f)[d \otimes f]$
$\mathcal{L}_3 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[f \otimes d]\ \text{J}\ (a, b, e)e$

Exemple. Contrainte 4

Contrainte 4 : S-langage

E et D repartent en même temps :

$$\mathcal{C}_4 = [e \otimes d](e, d)$$

Contraintes 4 : diagramme de Hasse

$$\begin{matrix} \text{d} \;— \\ \text{e} \;— \end{matrix}\begin{pmatrix} d \\ e \end{pmatrix}$$

S-langage résultant

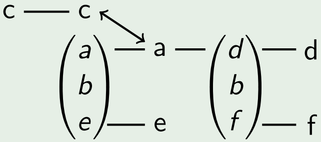
$\mathcal{L}_4 = \mathcal{L}_3 \text{ J } \mathcal{C}_4 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[d \otimes f] \text{ J } (a, b, e)e \text{ J } [e \otimes d](d, e)$

$\mathcal{L}_4 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[(d, e) \otimes f]$

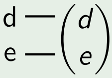
Exemple. Satisfaction de C1 & C2 & C3 & C4

$\mathcal{L}_4 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[(d, e) \otimes f]$

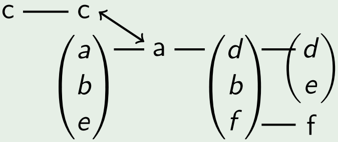
Etat C1 & C2 & C3



Contraintes 4



Etat C1 & C2 & C3 & C4



S-langage résultant

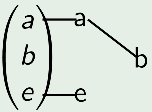
$\mathcal{L}_4 = \mathcal{L}_3$ J $\mathcal{C}_4 = [[c\vec{c} \otimes (a, b, e)]a(d, b, f)[d \otimes f]$ J $(a, b, e)e]$ J $[e \otimes d](d, e)$

$\mathcal{L}_4 = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[(d, e) \otimes f]$

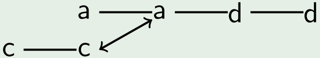
Exemple.
Calcul direct

$$\mathcal{U} = aa \otimes bb \otimes cc \otimes dd \otimes ee \otimes ff$$

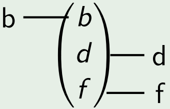
Contrainte 1



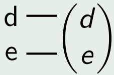
Contrainte 2



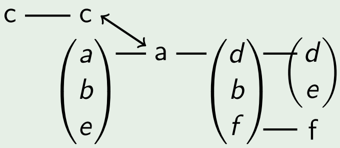
Contrainte 3



Contrainte 4



Jointures

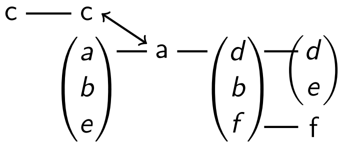


S-langage résultant

$$\mathcal{L} = \mathcal{U} \bowtie \mathcal{C}_1 \bowtie \mathcal{C}_2 \bowtie \mathcal{C}_3 \bowtie \mathcal{C}_4$$
$$\mathcal{L} = [c\vec{c} \otimes (a, b, e)]a(d, b, f)[(d, e) \otimes f]$$

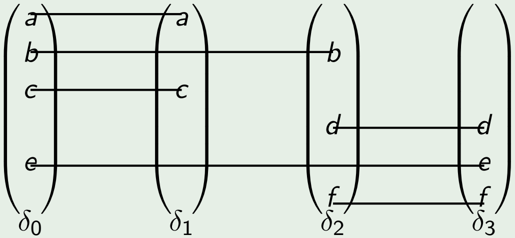
Le logiciel SLS [Irène Durand, Labri] calcule la jointure des contraintes.

Exemple.
Interprétation du résulat



$\mathcal{L}_4 = [c\vec{c} \otimes (a,b,e)]a(d,b,f)[(d,e) \otimes f]$
Il y a 24=$(5+3) \times 3$ S-mots contenant entre 5 et 7 S-lettres, de hauteur maximale comprise entre 3 et 4.
Le S-mot le plus court est $(a,b,c,e)(a,c)(d,b,f)(d,e,f)$. Il utilise le minimum de voies car il est dangereux de faire arriver sur une même voie un train en même temps que le départ d'un autre : B, D et F doivent être sur des voies différentes, et il faut une voie supplémentaire pour E.
Il faut donc au moins 4 voies pour satisfaire les contraintes

chronique



Quelques autres applications des S-langages

- évolution des cycles de vie en systèmes d'information (avec S. Jungjariyannon)
- gestion de séries temporelles (avec S. Jungjariyannon)
- temporalité et aspectualité en linguistique
- modélisation des systèmes de temps verbaux
- navigation textuelle (avec J.-L. Minel et D. Battistelli : TAL)
- modélisation de réseaux génétiques - cas du bacillus subtillis (biologie) (avec Nicolas Turenne- INRA)
- systèmes de partitions interactives (avec Antoine Allombert -Labri/Ircam)

Quelques références

- le logiciel SLS par Irène Durand : http://dept-info.labri.u-bordeaux.fr/ idurand/SLS/
- Durand & Schwer : *A tool for reasoning about qualitative temporal information with S-words and S-languages*, Journal of Universal Computer Science (2008) 14 (20) 3282-3306.
- S-langage et combinatoire : *S-arrangements avec répétitions*, Comptes Rendus de l'Académie des Sciences de Paris, Série I 334 (2002) 261–266.
- S-Univers et treillis : Autebert & Schwer : *On generalized Delannoy Paths*, SIAM Journal on Discrete Mathematics 16 (2) (2003) 208–223.
- Turenne & Schwer : *Temporal representation for gene networks: towards a qualitative temporal data mining* Int. J. Data Mining and Bioinformatics 2 (1), 2008, 36–53.