Représentation des connaissances : Introduction aux Graphes Conceptuels

Bernard ESPINASSE
Professeur à l'Université d'Aix-Marseille

Plan

- Définition d'un Graphe Conceptuel
- Types de concepts et relations dans les Graphes Conceptuels
- Contexte dans les Graphes Conceptuels
- Opérations dans les Graphes Conceptuels
- Forces et faiblesses des Graphes Conceptuels

Graphes Conceptuels (GC) : introduction

Les graphes conceptuels (GC) :

- ont été proposés par Sowa (1984), complété en 1988, puis 1992
- ils sont basés sur les graphes existentiels de Pierce
- de nombreux travaux de par le monde (équipe du LIRMM, Montpellier)
- ils constituent une notation graphique pour la logique
- ils sont plus formalisés que les réseaux sémantiques (dispose d'une sémantique formelle)
- ils sont beaucoup utilisés dans le traitement des langues naturelles

Différents niveaux dans le formalisme :

- Graphes simples
- Graphes imbriqués
- Graphes de règles

Représentation des Graphes Conceptuels

- Affichage graphique - DF (Display Form)
- Lecture linéaire – LF (Linear Form)
- CGIF (Conceptual Graph Interchange Format)

Définition d'un Graphe Conceptuel (GC)

Un GC est un graphe bipartite orienté

- Bipartite : 2 sortes de nœuds : **concept** et **relation**
- Les nœuds sont liés par des **arcs orientés**
- Un **arc** lie toujours un **concept** à une **relation**
- Un **nœud concept** peut être **isolé** (non rattaché)

Exemples :

« **le chat est sur le tapis** »

Forme graphique (Display Form – DF) :

2 concepts : chat et tapis et 1 relation : sur

Forme linéaire (Linear Form – LF) :

```
[chat]->(sur)->[tapis]
```


Définition des concepts dans les GC

Définition intentionnelle : Treillis de types :

- Ensemble de noms de concepts : Ec
- 2 éléments spéciaux maximaux :
  - le type « **universel** » (tout) : noté T (type de « toute chose »)
  - le type « **absurde** » (rien) : noté ⊥ (type impossible)
- une relation d'ordre partiel sur cet ensemble de types (spécialisation de types)
- pour un couple de concepts on peut définir un sur-type-commun-minimal et un sous-type-commun-minimal

Définition extensionnelle :

- a chaque type t de Ec on associe un ensemble d'objets [I(t)] : les référents possibles de t dans le monde
- un concept est représenté par un couple [type, référent] :
  - ref = * : concept générique (par défaut)
  - ref = #i : concept individuel
  - ref = @ : mesure

Exemple :

- `[personne : *]` = concept générique (un homme)
- `[personne : #123]` = concept individuel (cet homme)
- `[personne : Paul]` = concept individuel, nom propre
- `[hauteur: @173]` = concept mesure (hauteur = 1,73m)

Définition des relations dans les GC

Les **relations** permettent de représenter des propositions portant sur des concepts

Une relation est définie par 2 éléments (r, a) :

- r: nom de la relation
- a: **arité** ou **valence** (nb d'argument) de la relation = entier n donnant le nombre d'arc (1= monoadique, 2= dyadique, … n-adique)

On peut définir :

- **Signature** d'une relation : ensemble de n types de concepts
- Valence et signature sont fixés par le type de relation

Nature de relations :

- **Fonction** avec un argument ou plus dont le domaine de valeur est {vrai, faux}
- **Relation** : prédicat binaire ou plus
- **Propriété** : prédicat unaire
- **Algorithmes** et **inférences** selon leur type

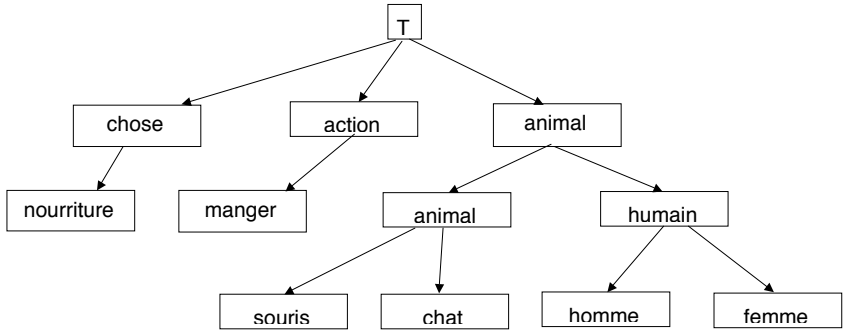
Les ensembles dans les GC

Type d'ensemble	Interprétation	Forme linaire
Concept générique	un homme	`[homme : *]`
Concept individuel	cet homme	`[homme : #123]`
Nom propre	Paul	`[homme : Paul]`
Mesure	Hauteur = 1,73 m	`[hauteur : @1,73]`
Ensemble extensif	Paul, Alain, Max	`[homme : Paul, Alain, Max]`
Ensemble **générique**	Plusieurs hommes	`[homme : {*}]`
Cardinal imbriqué	4 hommes	`[homme : {*}@4]`
Définition partielle	Paul, Alain, d'autres	`[homme : {Paul, Alain, *}]`
Ensemble **distributif** (Dist)	4 hommes (ont lu ces 2 livres)	`[homme : Dist{*}@4]`
Ensemble **respectifs** (Resp)	Paul, Alain (sont les maris respectifs de Alice, Marie)	`[homme : Resp {Paul, Alain}@2]`

Hiérarchie des types

Hypothèse sur la structure des connaissances stockées dans le GC :

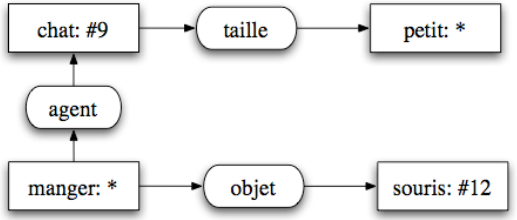
L'ensemble de tous les types (concepts) représentés est partiellement ordonné :



Exemple 1 de GC

Langage naturelle : « le petit chat mange la souris »

DF (display form) :



- 4 concepts: [manger :*], [chat: #9], [petit: *], et [souris : #12].
- 3 relations conceptuelles : (agent), (objet), (taille).

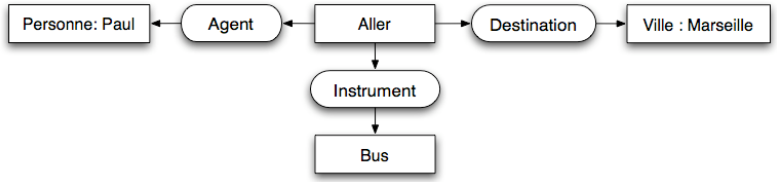
LF (linear form) :

```
[chat :#9]- (taille)->[petit :*]
[manger:*]- (objet)->[souris :#12]
            (agent)->[chat : #9]
```

Exemple 2 de GC

Langage naturelle : « Paul va à Marseille en bus »

DF (display form) :



- 4 concepts: [Aller], [Personne: John], [Ville: Marseille], et [Bus].
- 3 relations conceptuelles : (Agent) relie [Aller] à l'agent John, (Destination) relie [Aller] à la destination Marseille, et (Instrument) relie [Aller] à l'instrument bus.

LF (linear form) :

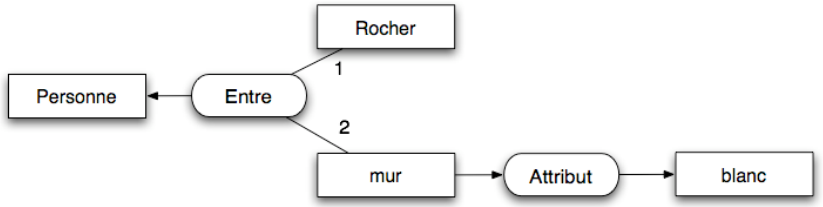
```
[Aller]-
   (Agent)->[Personne: Paul]
   (Destination)->[Ville: Marseille]
   (Instrument)->[Bus].
```

CGIF form : `[Aller: *x] [Personne: Paul *y] [Ville: Marseille *z] [Bus: *w] (Agent ?x ?y) (Destination ?x ?z) (Instrument ?x ?z)`

Exemple 3 de GC

Langage naturelle : « Une personne est entre un rocher et un mur blanc »

DF (display form) :



- valence de la relation Entre = 3
- signature de la relation Entre = (Entité, Entité, Entité)
-

LF (linear form) :

```
[Personne]<-(Entre)-
          <-1-[Rocher]
          <-2-[Mur]->(Attribut)->[Blanc].
```


Types de concepts

Types de concepts :

- **primitifs** : juste une étiquette et un positionnement dans la hiérarchie de types (treillis)
- **défini** : la définition est données par une λ-expression
- **prédéfinis** :
  - type **universel** note : T : type de « toute chose »
  - type **absurde** noté : ⊥ : type impossible
- **hiérarchie :** ordre partiel donné par la relation de subsomption (spécialisation de types) :

  ≥ : super-types de

  > : strictement super-type de

  ≤ : sous-type de

- **Exemple** :

```
Personne < Entité vivante
Directeur = [Personne : λ] -> (Diriger) -> [Groupe]
```


Types de relations

Hiérarchie des relations :

- **Hiérarchie** : ordre partiel donné par la relation de subsomption (spécialisation des types)
- **primitifs** : juste une étiquette et un positionnement dans la hiérarchie
- **défini** : la définition est données par une λ-expression

Exemple :

- un nom est une désignation :

```
nom < designation
```

- X vit à Y = « personne X habite à l'endroit Y » :

```
[Relation : VivreA] -> (Def) -> [LambdaExpression : Personne : λ1]
<- (Agent) <- [Habiter] -> (Emplacement) -> [Lieu : λ2 ]]
```

λ-expression

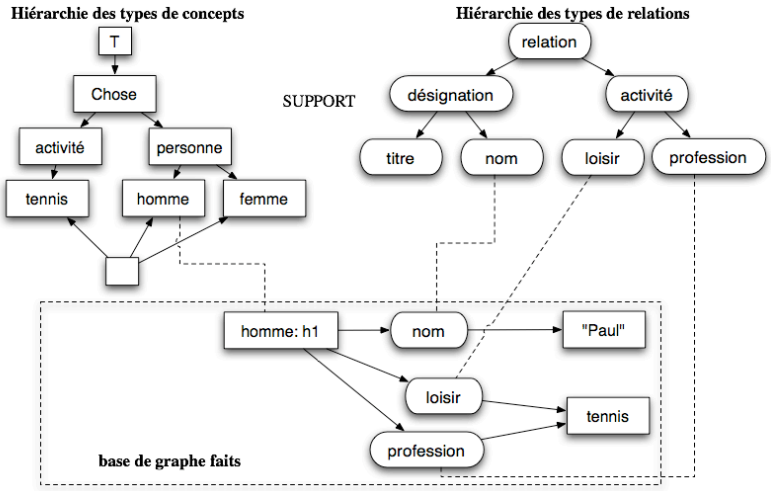
Lambda calculus [Alonzo Church 1941) :

$f(x) = 2x^2 + 3x - 2$ $f = \lambda\ x\ (2x^2 + 3x - 2)$

- intension/extension : égalité et définition
- faire des définitions formelles (interchangeables)
- une lambda expression n-adique est définie par un graphe (appelé le corps de l'expression) contenant n paramètres formels , λ_2, …, λ_n
- signature de l'expression = ensemble (t_1, t_2, t_3, …, t_n) où t_i est le type de concept du paramètre λ_i.


```
(lambda (personne *x, Lieu *y) [Habiter *z] (Agent ?z ?x)
  (Emplacement ?z ?y))
```


Types de concepts et de relations



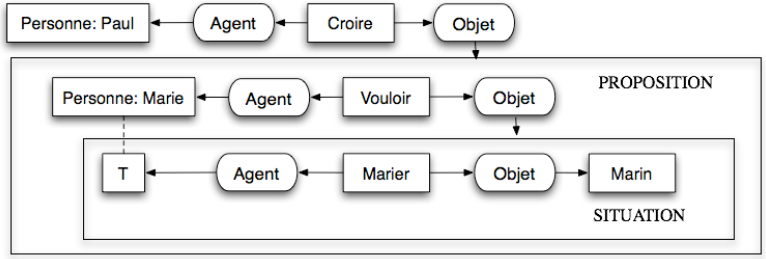
Contexte dans les GC

Le contexte = concept désignant un graphe non vide

Les contextes peuvent être imbriqués

Exemple :

Phrase : « Paul croit que Marie veut se marier avec un marin »



Forme linéaire :

```
[Personne: Paul]<-(Agent)<-[Croire]->(Objet)-
 [Proposition: [Personne: Marie *x]<-(Agent)<-[Vouloir]->(Objet)-
     [Situation:  [?x]<-(Agent)<-[Marie]->(Objet)->[Marin] ]].
```


Opérations dans les GC : dérivations

- **RECOPIE :** Produit un graphe identique au graphe de départ
- **RESTRICTION DE RÉFÉRENCE** : consiste à remplacer un concept *générique* par un concept de même type dont la référence est un élément de I(*t*)
- **RESTRICTION DE TYPE** : consiste à remplacer un concept *individuel* de type *t* et de référent I(*s*), *s* étant un sous-type de *t*, par un concept de même référence et de type *s*
- **SIMPLIFICATION** : consiste à supprimer des informations redondantes dans un graphe
- **JOINTURE** : la jointure de 2 graphes possédant un même concept (type et référent) est le graphe obtenu en attachant à ce concept toutes les relations qu'il a dans les 2 graphes de départ
- **JOINTURE MAXIMALE** : Le sous-graphe commun maximal est tel que les concepts qui y sont impliqués sont les plus grandes spécialisations communes des concepts correspondants des graphes de départ (pas forcément unique)
- **RELATION D'ORDRE PARTIEL** sur les graphes : u dérive de v si u peut être obtenu à partir de v par une série d'applications des règles de dérivation (u est une spécialisation de v)

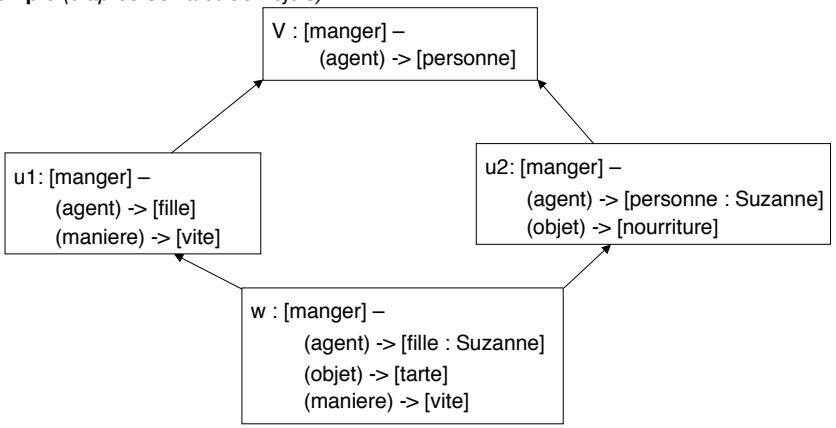
Graphes dérivés d'un ensemble B de graphes = informations, potentiellement vraies, que l'on peut tirer de B

Opérations dans les GC : jointure (ou généralisation)

Soit *u1*, *u2*, *v* et *w* des graphes conceptuels.

- Si *u1*≤*v* et *u2*≤*v*, alors *v* est appelé ***généralisation commune*** de u1 et *u2*.
- Si *w*≤*u1* et *w*≤*u2*, alors *w* est appelé ***spécialisation commune*** de *u* et *u*.

Exemple *(d'après Sowa et Cornéjols) :*



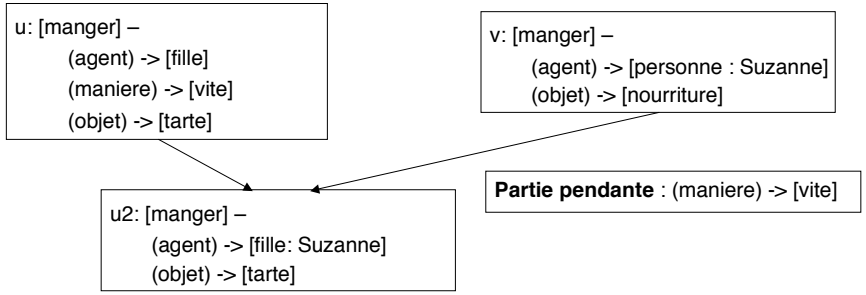
Opérations dans les GC : projection (1)

Théorème : Pour tous graphes conceptuels *u* et *v*, où *u*≤*v*, il existe une opération π telle que :
π : *v* --> *u*, où π*v* est un sous-graphe de *u* appelé **projection** de *v* dans *u*.

L'opérateur π a les propriétés suivantes :

- Pour chaque concept *c* dans *v*, π*c* est un concept dans π*v* où *t*(π*c*) = *t*(*c*). Si *c* est un concept individuel, alors ref(*c*) = ref(π*c*).
- Pour chaque relation conceptuelle *r* dans *v*, π*r* est une relation conceptuelle dans π*v* où *t*(π*r*) = *t*(*r*). Si le ième arc de *r* est relié à un concept *c* dans *v*, le ième arc de π*r* doit être relié à π*c* dans π*v*.

Exemple 1 *(d'après Sowa et Cornéjols) :*



Opérations dans les GC : projection (2)

Exemple 2 *(d'après Sowa et Cornéjols)* :

u: [aimer] –
(agent) -> [homme : Jules] <- (enfant) <- [homme : Jean]
(objet) -> [femme : Marie] <- (enfant) <- [homme : Jean]

v: [personne :*x] –> (enfant) -> [personne: *y]

2 projections :

πv: [homme: Jules] <- (enfant) <- [homme: Jean]

πv': [femme: Marie] <- (enfant) <- [homme: Jean]

Partie pendante :

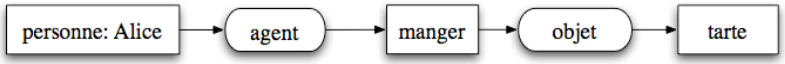
[aimer] -> (agent)
-> (objet)

Exemples d'opérations sur les GC (1)

Soit les *graphes canoniques* suivants :

"Une fille mange vite" - graphe (1):

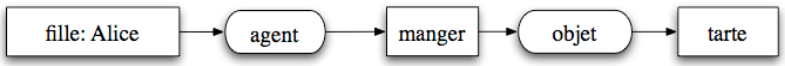
"une personne, Alice, mange une tarte" - graphe (2):



1- Restriction dans (2) de « personne » à « fille » :

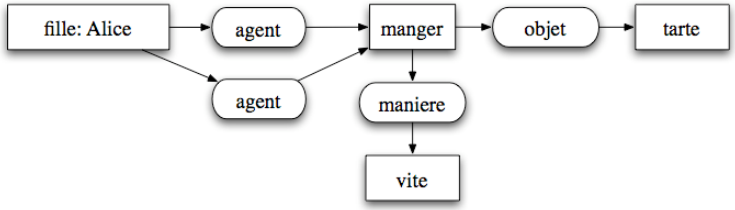
Vérification que :

- fille < personne
- fille : Alice est vrai

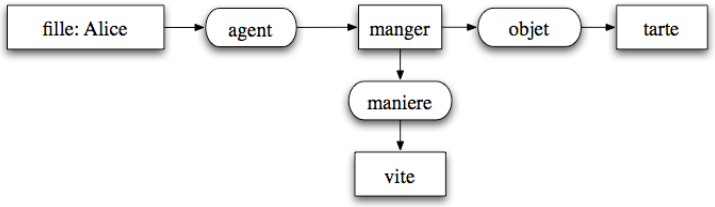


Exemples d'opérations sur les GC (2)

2- Jointure entre les graphes (1) et (2):
la jointure se fait sur les concepts identiques (après la restriction précédente) – graphe (4):



Simplification du graphe (4) :



« une fille, Alice, mange vite une tarte »

Conclusion sur les Graphes Conceptuels

Forces :

- C'est un **très puissant formalisme de représentation de la connaissance** :
  - doté d'une **représentation graphique**
  - disposant d'une **sémantique formelle** (équivalence avec la logique)
- **Nombreux travaux en cours** (sur les opérateurs, introduction de notions de vraisemblance, de précision, …)
- Très éfficace en analyse du **langage naturel**
- Présence de **nombreux outils** : CoGiTant, Notio (API Java), CharGer (éditeur GC), …

Faiblesses :

- Problèmes de disjonction
- Problèmes de négation
- Problème d'imbrication de quantificateurs