Vers une Planification basée sur une Théorie Constructive des Types en Logique Intuitionniste

Journées IAF - AFIA

Richard Dapoigny Patrick Barlatier

Polytech'Savoie
BP 80439
74944 ANNECY LE VIEUX cedex - France

Grenoble 2-3 juillet 2007

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Sommaire

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Sommaire

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Les limites de la planification classique

- Faible expressivité des connaissances du domaine.
- Diminuer la complexité des planificateurs.
- Dans les applications réelles, la recherche de plans optimaux est un problème difficile à traiter [Pollock06].
- Dépasser les limites imposées par la logique de premier ordre.
- En environnement partiellement observable, la logique doit être capable de représenter les interactions entre action et connaissance.

La théorie des types a été explorée pour comparer des approches de planification [Fox-Long98] et pour modéliser l'interaction avec l'utilisateur [Fox-Long01]

- Approfondir et améliorer cette approche.

Objectifs

- Théorie Intuitionniste des types $\rightarrow$ bien implantée en Traitement des Langues Naturelles ainsi qu'en théorie des langages.
- Objectif : étendre son utilisation à la planification pour la formalisation des connaissances du domaine.
- **Intérêt** : fournir une théorie
  - Permettant l'unification entre les formalismes de représentation et de raisonnement en IA (un même modèle pour dialoguer avec l'utilisateur, pour gérer les ontologies et pour raisonner sur les actions et leurs contextes).
  - Représentant une simplification notable par rapport aux théories existantes (à la fois dans la structure et dans les composants logiques utilisés).
  - Améliorant la décidabilité des modèles produits.

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Vers une planification utilisant la théorie Intuitioniste des types (ITT)

- Deux composants fondamentaux nécessaires à la planification :
  - la représentation des connaissances requise
  - la logique sous-jacente au planificateur.
- Idée : unifier ces composants en fournissant un système de représentation des connaissances évolué (ontologie) associé à une logique suffisamment expressive (HOL).
- Une stratégie de planification doit être vue comme un type inductif générant une famille de théorèmes correspondant aux plans recherchés.

Planification en Théorie Intuitioniste des Types

Construction des plans

- Theorem proving utilisant l'isomorphisme de Curry-Howard.
- Sous-typage à grain élevé.

Validation des plans à l'exécution

- Type-checking.
- Sous-typage.

Sommaire

Types de base

Spécifier :

- La modélisation des types de base et leur relations par une ontologie.
- La représentation du contexte de l'action par un enregistrement à types dépendants ou *Dependent Record type* (DRT).
- La représentation de l'action et de ses effets par une famille de fonctions indexées.
- Les règles de sous-typage.
- Les règles de chainage des DRTs.

Représentation du contexte en IA

Approches logiques

- L'approche de McCarthy qui est définie dans le cadre du *Situation Calculus* avec règles de lifting [McCarthy93].
- Les systèmes Multi-contextuels qui reposent à la fois sur le principe de localité et sur le principe de compatibilité gérant les relations entre les différents contextes [Giunchi92].
- L'approche basée sur une théorie des types qui utilise trois constructions syntaxiques primitives (i.e., identité, application fonctionelle et lambda abstraction) [Thom99].

Les Principales limitations

- Ces approches montrent certaines difficultés à gérer l’aspect intentionnel (excepté la troisième).
- La plupart d’entre elles utilise la modalité comme une solution. Toutefois, la logique modale qui repose sur un concept "essentialiste" induit des axiomes arbitraires [Girard03].
- Elles ont un pouvoir d’expression limité inhérent à la logique de premier ordre.
- Elles manquent de capacités dynamiques.
- Elles ont une difficulté pour exprimer la validation partielle.

Sommaire

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Qu’est-ce que la logique Intuitionniste?

- Elle possède la propriété d’existence : seuls les concepts mathématiques pouvant être démontrés ou construits sont légitimes.
- Affirmer qu’un objet ayant certaines propriétés existe, c’est affirmer être capable de construire un objet avec ces propriétés.

Pourquoi la théorie des Types?

- C’est la base d’importants travaux en logique et en IA (theorem provers tels que Coq, LEGO, ...), les langages fonctionnels (Lisp, Caml, Haskell, ...), les logiques (Logique linéaire, théorie des types de Martin Löf, ...).
- La théorie des Types offre un riche système de typage dans lequel toute propriété en logique des prédicats peut être interprétée en tant que type.
- La logique intuitionniste d’ordre supérieur (avec typage) est valide et complète [Coniglio01].

Les apports de la Théorie Intuitionniste des Types

- Le paradigme "Proofs-as-programs": associe une preuve constructive à un programme réalisant la formule (How69).
- Fournit un support pour les types dépendants.
- Garantit la décidabilité de tout jugement : ITT est fonctionnellement décidable [Valent99].
- Utilisée en Traitement des Langues Naturelles et pour les certifications de programmes.

Sommaire

Les Types Dépendants

- Types Dépendants : types exprimés en fonction de données.
- En quoi sont-ils utiles? Par leur aptitude à classifier les données par des critères.

Principaux avantages :

- Capacité d'exprimer des relations indexées.
- Meilleure flexibilité que les systèmes de types indépendants.
- Continuum de précision s'étendant d'une simple assertion jusqu'à une spécification complète d'un problème).
- Décidabilité [Coq05].

DRT : objets de première classe

Définition

Un enregistrement à types dépendants est une séquence de champs labellisés dans laquelle les labels l_i correspondent à certain types T_i, c-à-d que chaque champ successif peut dépendre des valeurs des champs le précédant:

- $< l_1 : T_1, l_2 : T_2(l_1) \ldots, l_n : T_n(l_1 \ldots l_{n-1}) >$,
- où le type T_i peut dépendre des labels précédents $l_1, \ldots, l_{i-1}$.

DRT : objets de première classe

Définition

Un enregistrement à types dépendants est une séquence de champs labellisés dans laquelle les labels l_i correspondent à certain types T_i, c-à-d que chaque champ successif peut dépendre des valeurs des champs le précédant:

- $< l_1 : T_1, l_2 : T_2(l_1) \ldots, l_n : T_n(l_1 \ldots l_{n-1}) >$,
- où le type T_i peut dépendre des labels précédents $l_1, \ldots, l_{i-1}$.

Sommaire

Principes de base

- Contextes en IA vus comme une liste ordonnée de définitions de types et de pré-suppositions.
- Contextes représentés par des Dependent Record Types (DRT) :
  - types de contextes (la potentialité).
  - objets contextes (la réalité).
- Identification partielle des objets contexte par une situation (c-à-d, objets de contexte sont *partie-de* situations).
- Les contextes, de même que les DRTs peuvent être étendus.
- Introduction d'une ontologie représentant les termes et les relations du domaine pour améliorer la ré-utilisabilité et la maintenance.

DRT : Objets de première classe

Example

Type de contexte :

$$\left\lceil \begin{array}{rcl} x & : & \mathit{Vehicle} \\ y & : & \mathit{RegistrationNumber} \\ l_1 & : & \mathit{GPSLongitude} \\ l_2 & : & \mathit{GPSLatitude} \\ t_e & : & \mathit{evTime} \\ t_m & : & \mathit{maxTime} \\ q_1 & : & \mathit{has\_identification}(x, y) \\ q_2 & : & \mathit{has\_Longitude}(x, l_1) \\ q_3 & : & \mathit{has\_Latitude}(x, l_2) \\ c_1 & : & \mathit{has\_lowerValueThan}(\mathit{evTime}, \mathit{maxTime}) \end{array} \right\rfloor$$

DRT : Objets de première classe

Example

Objet Contexte :

$$\begin{bmatrix} x = & truck \\ y = & 2678KX69 \\ l_1 = & 12.0987 \\ l_2 = & 67.2365 \\ t_e = & 2/11/06.11:33 \\ t_m = & 2/11/06.12:00 \\ q_1 = & p_1 \\ q_2 = & p_2 \\ q_3 = & p_3 \\ c_1 = & ct_1 \end{bmatrix}$$

DRT : Objets de première classe

Les jugements de types suivants sont valides :

- p_1 est une preuve de $has\_identification(truck, 2678KX69)$
- p_2 est une preuve de $has\_Longitude(truck, 12.0987)$
- p_3 est une preuve de $has\_Latitude(truck, 67.2365)$
- ct_1 est une preuve que l'instant d'évaluation est plus petit que la valeur limite $(2/11/06.11 : 33 < 2/11/06.12 : 00)$

Si l'un des composants du type contexte n'est pas peuplé, alors le type de contexte n'est pas validé.
DRTs décrivant un contexte $\rightarrow$ CRTs (Context Record Types).

Règles de Formation des CRTs

CRT vide

$$\frac{}{\Gamma \vdash <> : record - type}$$

Existence d'un tout : le CRT vide.

Extension de CRT

$$\frac{\Gamma \vdash R : record - type \quad \Gamma \vdash T : record - type \rightarrow type}{\Gamma \vdash < R, l : T > : record - type}$$

Etendre un CRT par un champ l de type T est encore un CRT.

Sous-typage des CRTs

- Un mécanisme de sous-typage dans la théorie à types dépendants est nécessaire à la réalisation des plans (enchainement des actions).
- La plupart des travaux introduisent soit des règles spécifiant les coercions [Luo99] soit un sous-typage centré sur les enregistrements à types dépendants [Bet00].

Sous-typage des CRTs

- Le sous-typage $\longrightarrow$ flexibilité
- Mais problème des coercions à actions non-intuitives $\longrightarrow$ nécessite la connaissance de toutes les coercions possibles pour un terme donné ainsi que leur effet précis (ingérable en pratique).
- Solution $\longrightarrow$ imposer des contraintes sémantiques sur les coercions.
- Adoption de ce sous-typage dans les enregistrements à types dépendants.

Sous-typage des CRTs

- Les coercions "oublient" la structure des enregistrements à types dépendants pour générer des enregistrements plus petits.
- Dans le domaine de l'*IA*, l'extension d'un contexte correspond au fait d'accumuler d'avantage d'information.
- Un contexte C est étendu (lifté) à un contexte C' ssi C' possède d'avantage d'information que C :

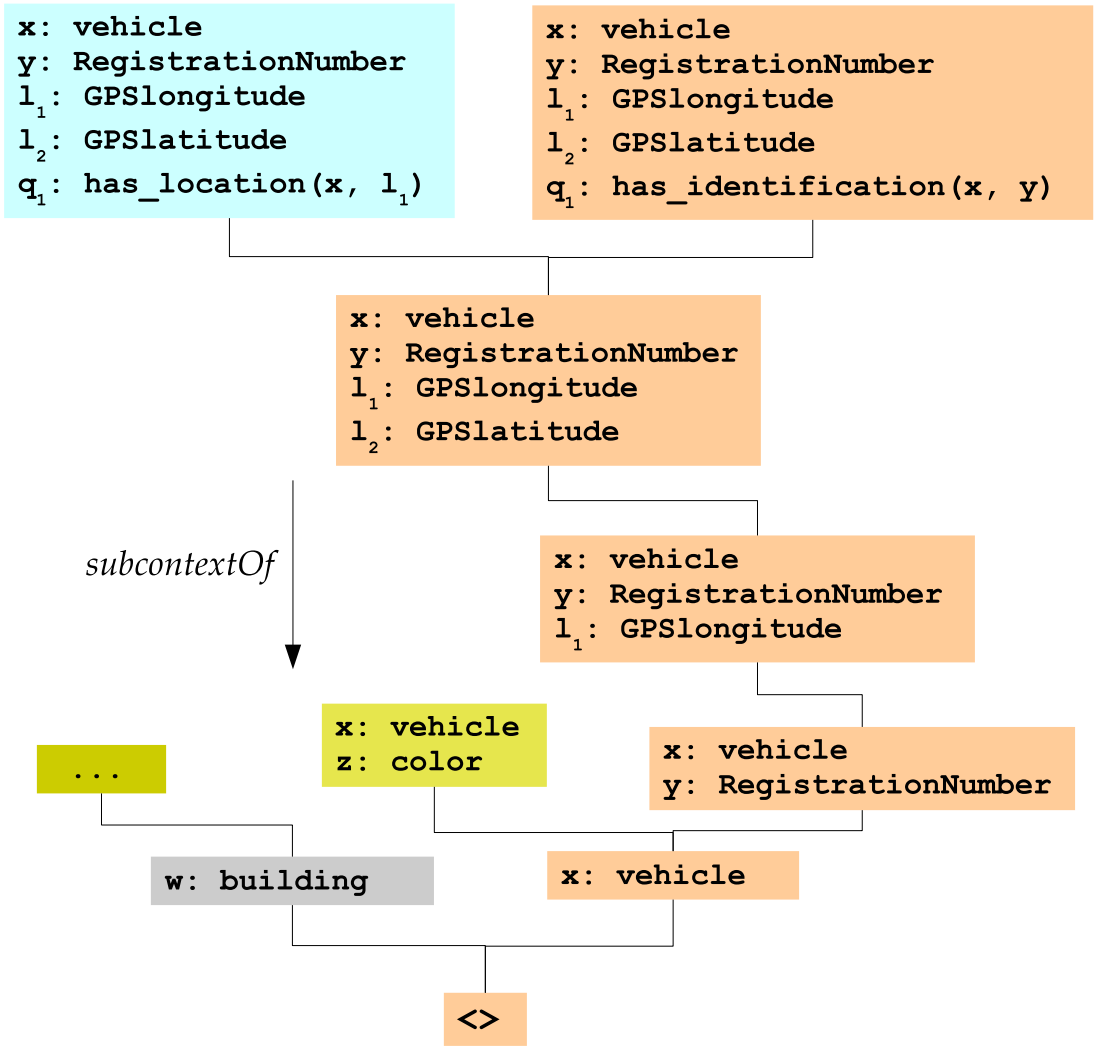
$$C' \sqsubseteq C \ \ si \ \ CRT(C) \subseteq CRT(C')$$

- C' est appelé sous-contexte de C.

Tout CRT est "partie-de" du CRT vide

$$\frac{R : record - type}{\Gamma \vdash R \sqsubseteq <>}$$

Mécanisme de Sous-typage



Sommaire

Définitions

L'action est décrite par une famille de types d'enregistrements $\mathcal{F}$ qui est une fonction d'enregistrements de type CRT vers des types d'enregistrements :

$$\lambda c : C.[a : action\_verb(\ldots, c.l_i, \ldots)$$

où *action_verb* est un verbe d'action, l_i un champ de c et a une proposition preuve de l'action effectuée.

- Action =
  1. conséquence de la présence d'un contexte
  2. pré-condition à l'existence des effets de cette action.
- Notion de point fixe : a est un point fixe pour $\mathcal{F}$ si $a : \mathcal{F}(a)$.

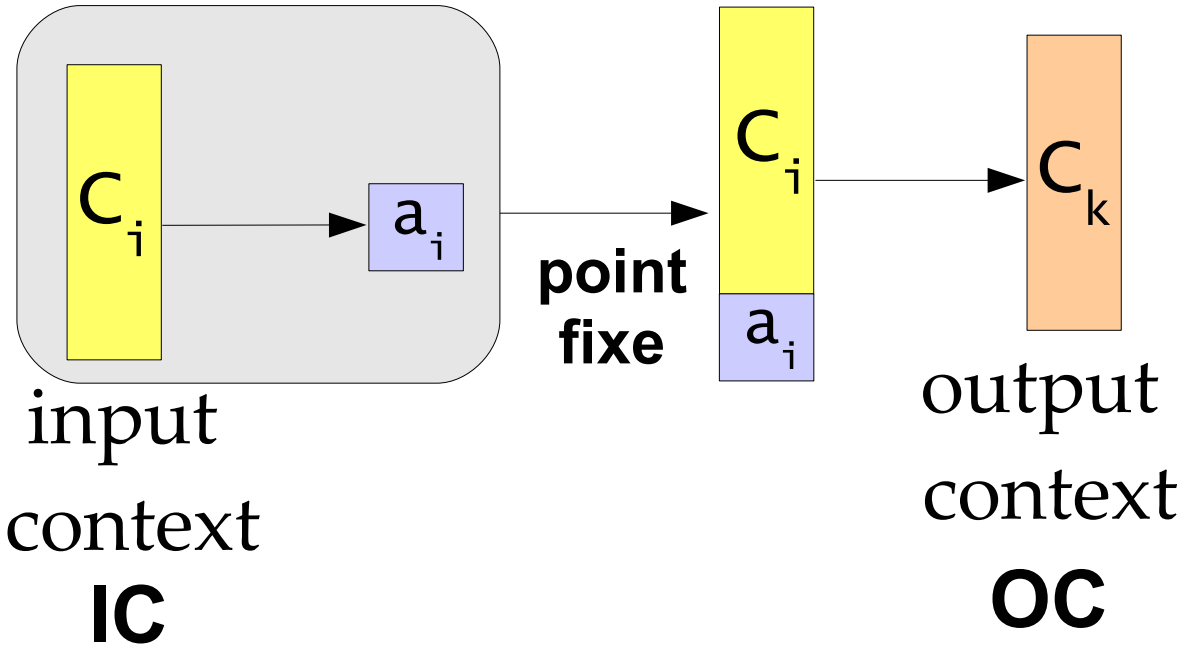
Definition des paires fonctionnelles

- But + effets (*OC* : vu comme un contexte de "sortie").
- *OC* est fonction d'un contexte possible (*IC*) et d'une action associée.

$$\lambda r : \mathcal{F}(\mathcal{A}). \left[\begin{array}{l} g : [g_1 : ... \\ e : \left[\begin{array}{ll} e_1 : & \ldots \\ e_2 : & \ldots \end{array} \right. \end{array} \right. \quad (1)$$

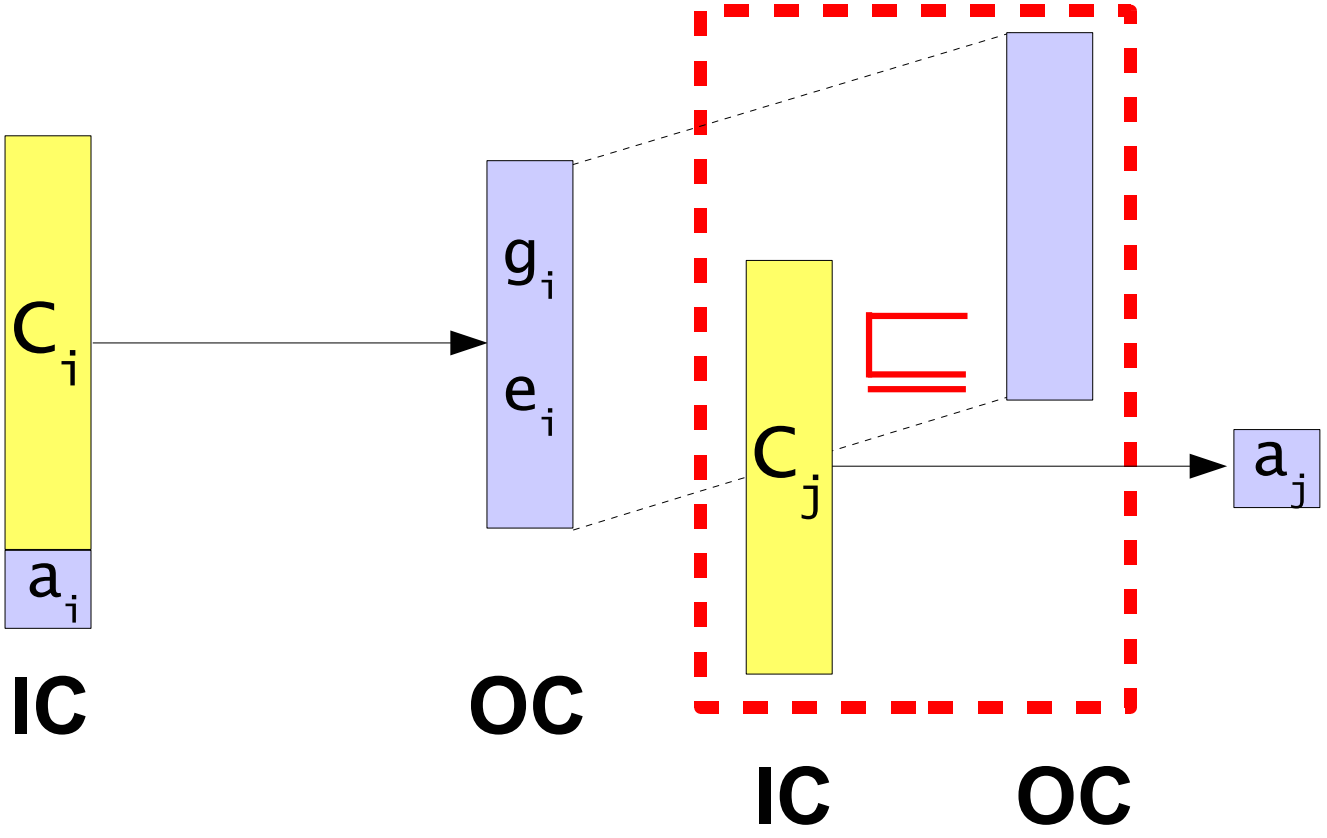
En résumé, si un contexte existe dans une situation donnée, alors l'action associée est exécutable et le but est réalisable.

Point fixe



Sommaire

Opérateur séquentiel



Opérateur séquentiel

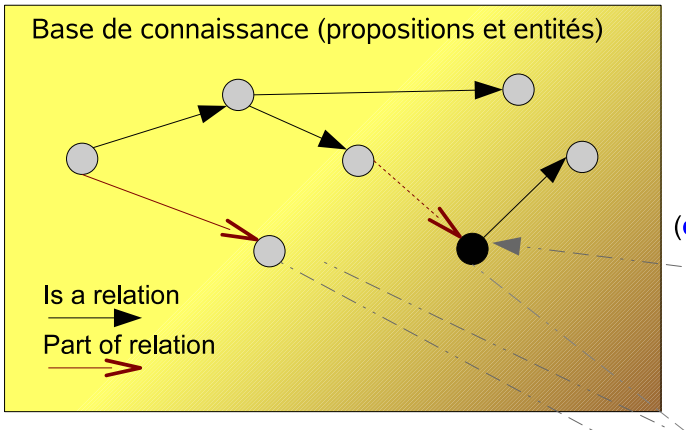
$$\frac{\Gamma \vdash \lambda r : \left[\begin{array}{l} IC_i \\ a_i : [... \end{array}\right. \quad .g : [OC_i : record - type \qquad \Gamma \vdash \lambda r' : \left[\begin{array}{l} IC_j \\ a_j : [... \end{array}\right. \quad .g' : \left[OC_j : record - type \qquad \Gamma \vdash IC_j \sqsubseteq OC_i}{\Gamma \vdash SEQ(a_i, a_j)}$$

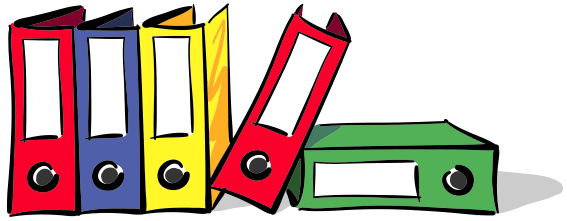
LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Application au Planning

- Algorithme de planification axé sur la causalité entre CRTs.
- Planification : progression dans laquelle la direction de recherche part de la situation initiale.
- Pas de recherche d'états possibles, mais recherche d'objets Contexte valides par vérification de types
- Recherche des chemins dans lesquels les noeuds sont des CRTs en relation de causalité.

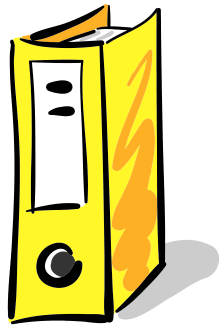
L'arbre de recherche

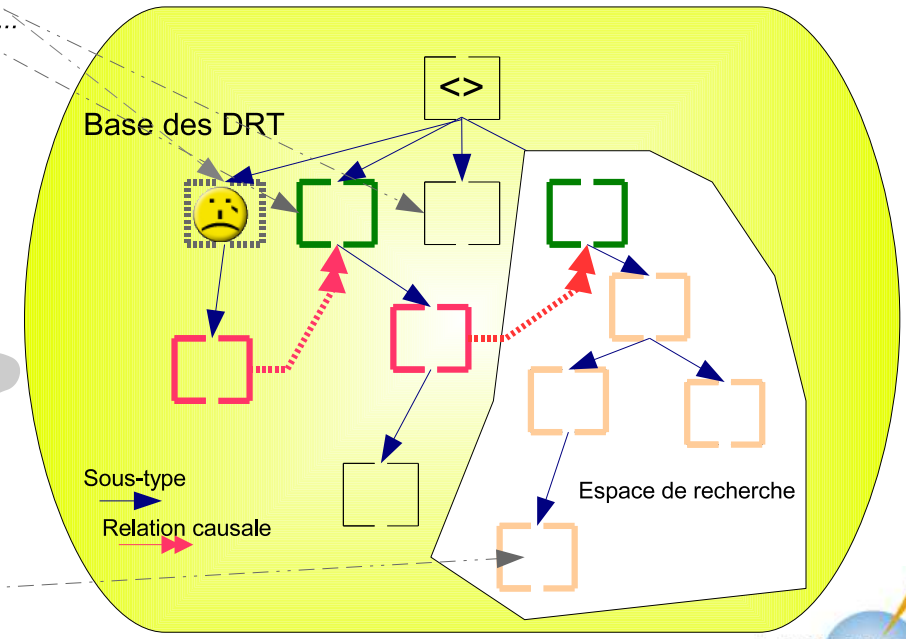




```
(defclass person ()
    ((name :accessor name :initarg :name)
     (at :accessor at:initarg :at)
     (has-usage :accessor has-usage
                :initarg :has-usage)
     (in :accessor in :initarg :in)))
```

Utilisée par...






```
(defclass drt ()
  ((proof :initform '() :initarg :proof :accessor proof)
   (list-of-labels :initform '() :initarg :list-of-labels :accessor list-of-labels)
   (direct-surtype :initform '() :initarg :direct-surtype :accessor direct-surtype)
   (list-of-direct-subtypes :initform '() :initarg :list-of-direct-subtypes :accessor list-of-direct-subtypes)
   (action-link :initform '() :initarg :action-link :accessor action-link)))
```

LISTIC Polytech'Savoie
laboratoire d'informatique systèmes, traitement de l'information et de la connaissance

Comparaison au modèle STRIPS

		Sémantique STRIPS	Sémantique basée sur ITT
Comparaison structurelle	Approche	Epistémologique	Ontologique
	Logique	Premier ordre	ITT
	Hypothèse du monde clos	Oui	Non
	Problème traité	Etat initial + buts + ensemble d'actions	Situation initiale + buts + paires CRT-but
	Grain	Prédicats	DRTs
	Représentation des actions	Préconditions + effets	DRT + fonctions
	Effets conditionnels	Non	Oui
Comparaison fonctionnelle	Mécanisme	- si precond(a)=TRUE dans l'état s alors activer a	- déterminer la situation initiale et le(s) contexte(s) valide(s) par type-checking
		- substitution de variables	-sélection de l'action a
		- détermination de l'état final s' par mise à jour de s avec les effets	-recherche du nouveau contexte par sous-typage
		- arrêt quand le but appartient à s'	- arrêt quand le but est un sous-type du contexte obtenu
	Traitement du frame probleme	Tous les prédicats de s absents des effets restent inchangés	Seules les propositions et les types prouvés sont valides à un instant donné

Principaux apports

- Haut niveau d'expressivité.
- L'introduction de types permet de factoriser les contextes et de réduire la complexité globale.
- Automatisation de la validation des contextes à l'exécution.
- Automatisation de l'acquisition des connaissances en planification.
- Incorporation de la validation partielle dans la logique.
- L'approche basée sur les types a un impact computationnel immédiat.

Des limitations

1. Les types sont prédéfinis. Toutefois, dans les environnements distribués, de nouveaux types peuvent être générés par fusion de connaissances à partir des autres composants.
2. Pas encore de version distribuée du système.
3. Gestion des inconsistances entre buts.

Travaux en cours

1. Optimiser l’algorithme de sous-typage.
2. Examiner les aspects temporels.
3. Fournir une interface utilisateur.
4. Etendre le modèle aux systèmes distribués.

Questions et discussion

Questions et discussion