f

Prog2 - Projet 1
Interprète LISP

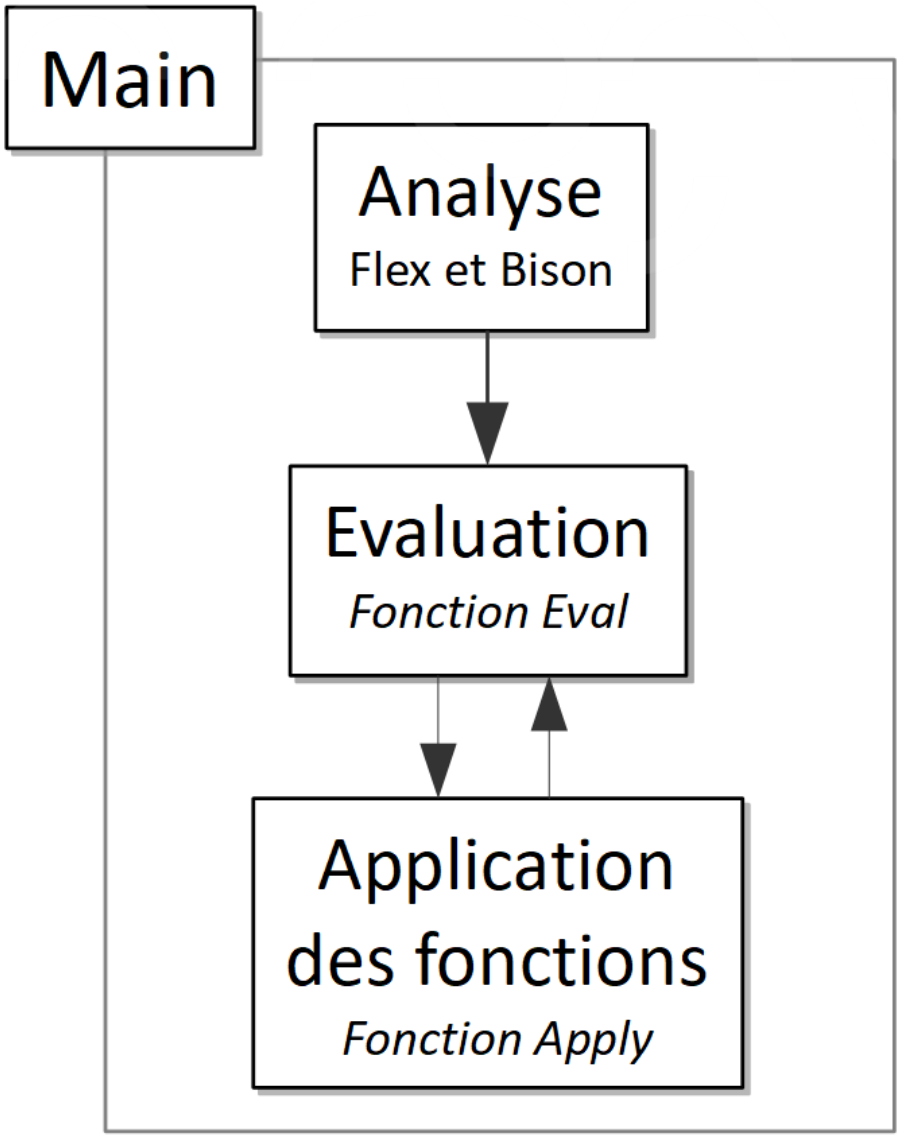
J. Cabrita C. Legrand-Lixon A. Lequen G. Mescoff

ENS Rennes

28 février 2018

Interprète LISP

- LISP (1958) : langage de programmation fonctionnel
- Interprète : programme exécutant dynamiquement un script non compilé


Main
Analyse
Flex et Bison
Evaluation
Fonction Eval
Application
des fonctions
Fonction Apply

Grammaire

Forme de Backaus-Naur

expression $\rightarrow$ *liste* | *nombre* | *symbole* | *string* | *nil*
liste $\rightarrow$ *(expression)*

$\rightarrow$ Lisp peut être décrit par des règles simples

Analyses lexicale et syntaxique

Flex : Analyse lexicale

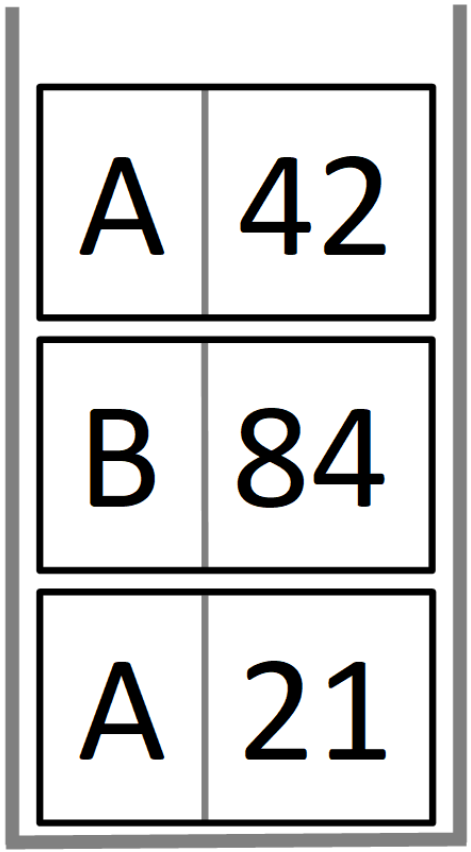
Transforme des chaînes reconnues par des expressions régulières en *tokens*

Bison : Analyse syntaxique

Convertit une suite de *tokens* en objets reconnus par l'interprète

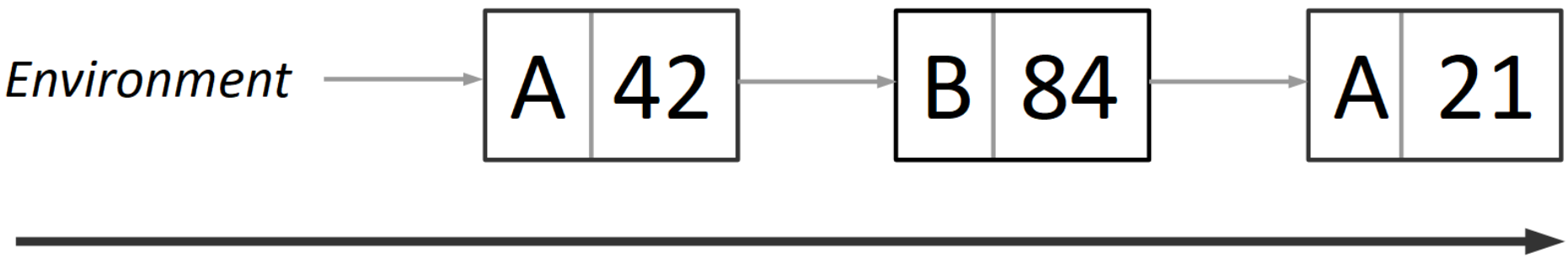
Première approche : liaisons dynamiques

- Structure de pile
- Liaisons jamais modifiées
- Liaisons masquées



Liste chaînée

- Facile à implémenter
- Recherche d'un élément assez longue



Idée : remplacer par une table de hachage

Fonction eval

Renvoie la valeur de l'expression passée

- Chercher les valeurs des symboles dans l'environnement
- Modifier l'environnement
- Identifier les sous-routines natives
- **Demander l'application des sous-routines**

Gestion des erreurs

Lève une exception en cas d'incohérence sémantique

Fonction apply

Applique une fonction à une liste d'arguments

- Vérifie si la liste commence par une fonction
- Applique la fonction aux arguments passés
- Retourne la valeur de l'expression évaluée

Remarques : Fonction apply

- Mutuellement récursive avec eval
- Ignore les arguments supplémentaires

Gestion des erreurs

Lève une exception si :

- Le premier élément n'est pas évalué en une fonction
- Trop peu d'arguments suivent

Implémentation des différentes sous-routines

- Sous-routines implémentées dans des fonctions propres
- Conventions de nommage et de prototypes

Ajout rapide de nouvelles fonctions natives

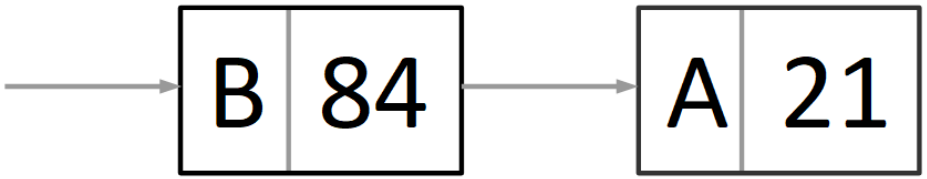
Exemples de sous-routines natives

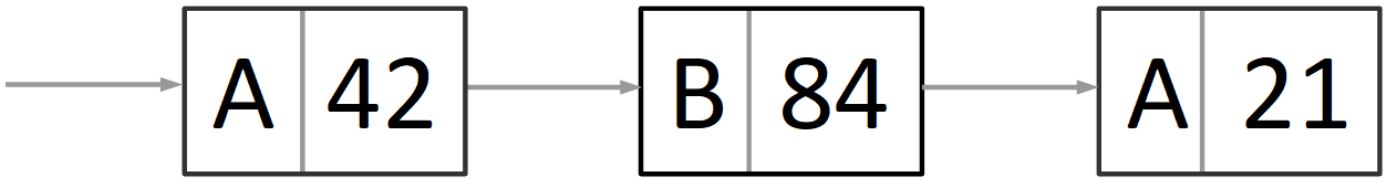
- $+$, $*$, $-$, ...
- lambda, quote, print, setq, newline...
- apply, eval
- progn, map
- printenv

Deux manières de modifier les liaisons

Dynamique : define

- Ajoute une nouvelle liaison à la liste
- Masque les liaisons précédentes

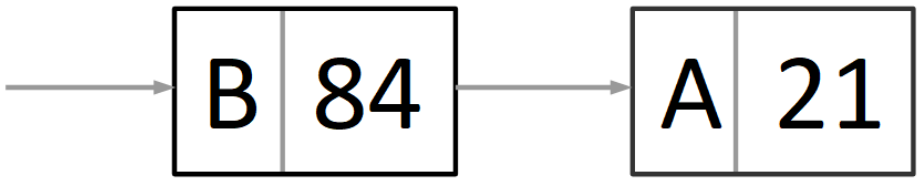


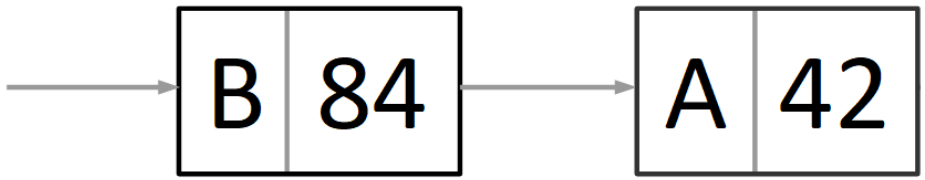


Deux manières de modifier les liaisons

Statique : setq

- Crée une nouvelle liaison si le symbole n'est pas encore lié
- Modifie la liaison la plus récente pour un symbole dans l'environnement courant





Closure

Capture (*snapshot*) de l'environnement au moment de la déclaration d'une lambda-expression

Implémentation

- Enregistre l'environnement à chaque symbole *lambda*.
- Pointeur vers un environnement

SWOT

- **Forces :** Organisation agile, planification
- **Faiblesses :** Gestion hasardeuse des tests et exceptions
- **Opportunités :** Meilleure compréhension de la gestion de l'environnement
- **Menaces :**