ALF

Arbre de syntaxe abstraite

Bibliographie pour aujourd'hui

ALF
language crunch

Keith Cooper, Linda Torczon, *Engineering a Compiler*

- Chapitre 4
  - 4.1 – 4.5
- Chapitre 5

Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, *Compilers: Principles, Techniques, and Tools (2nd Edition)*

- Chapitre 5
  - 5.1
  - 5.2
  - 5.3

Contenu

ALF
language crunch

- Types
- Arbre de syntaxe abstraite
- Analyse sémantique

Grace Hopper

- Américain
- Vassar College
- Yale
- Premier compilateur
- Langage A-0

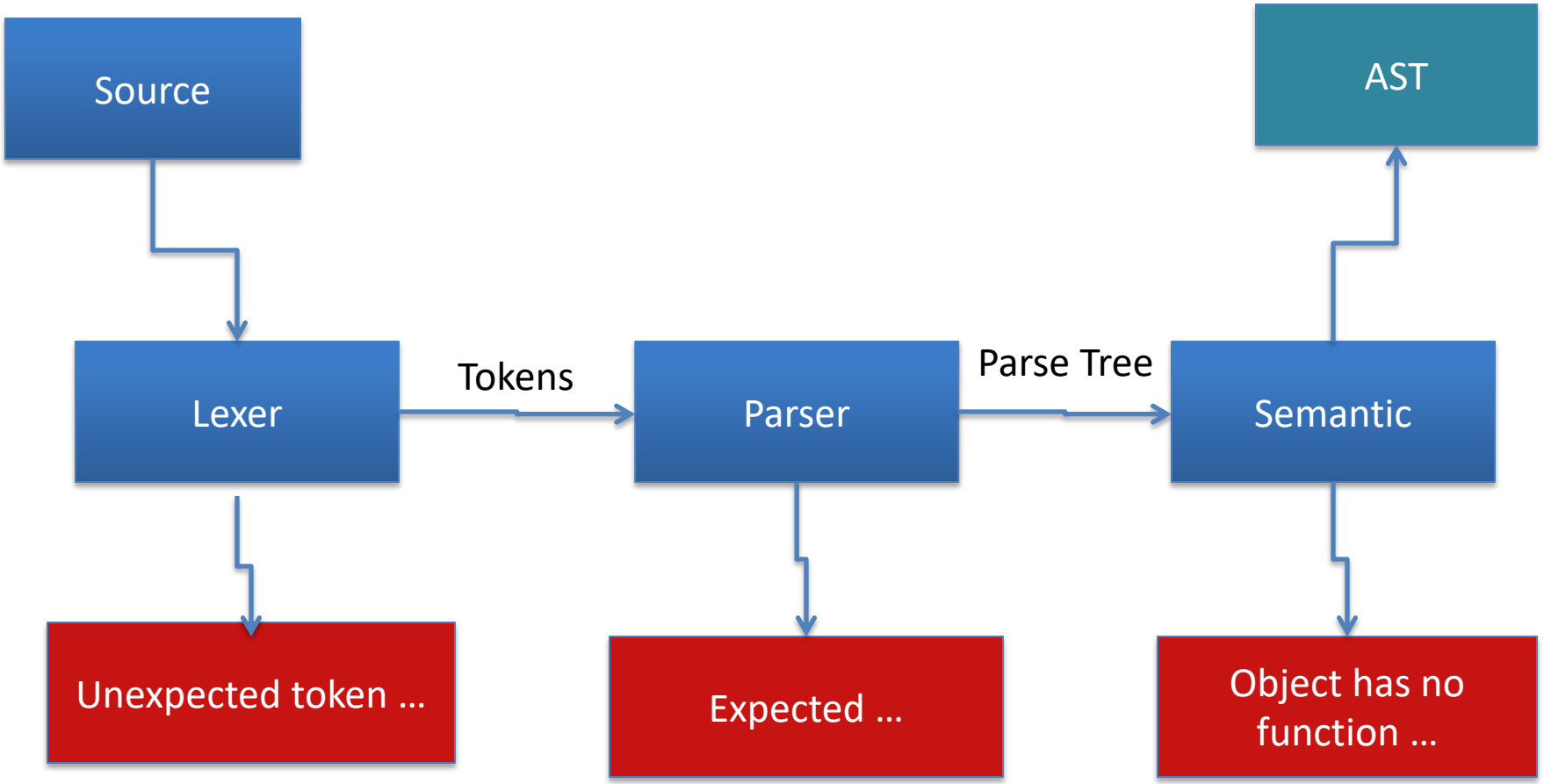
Slides

Partie de slides sont écrie par

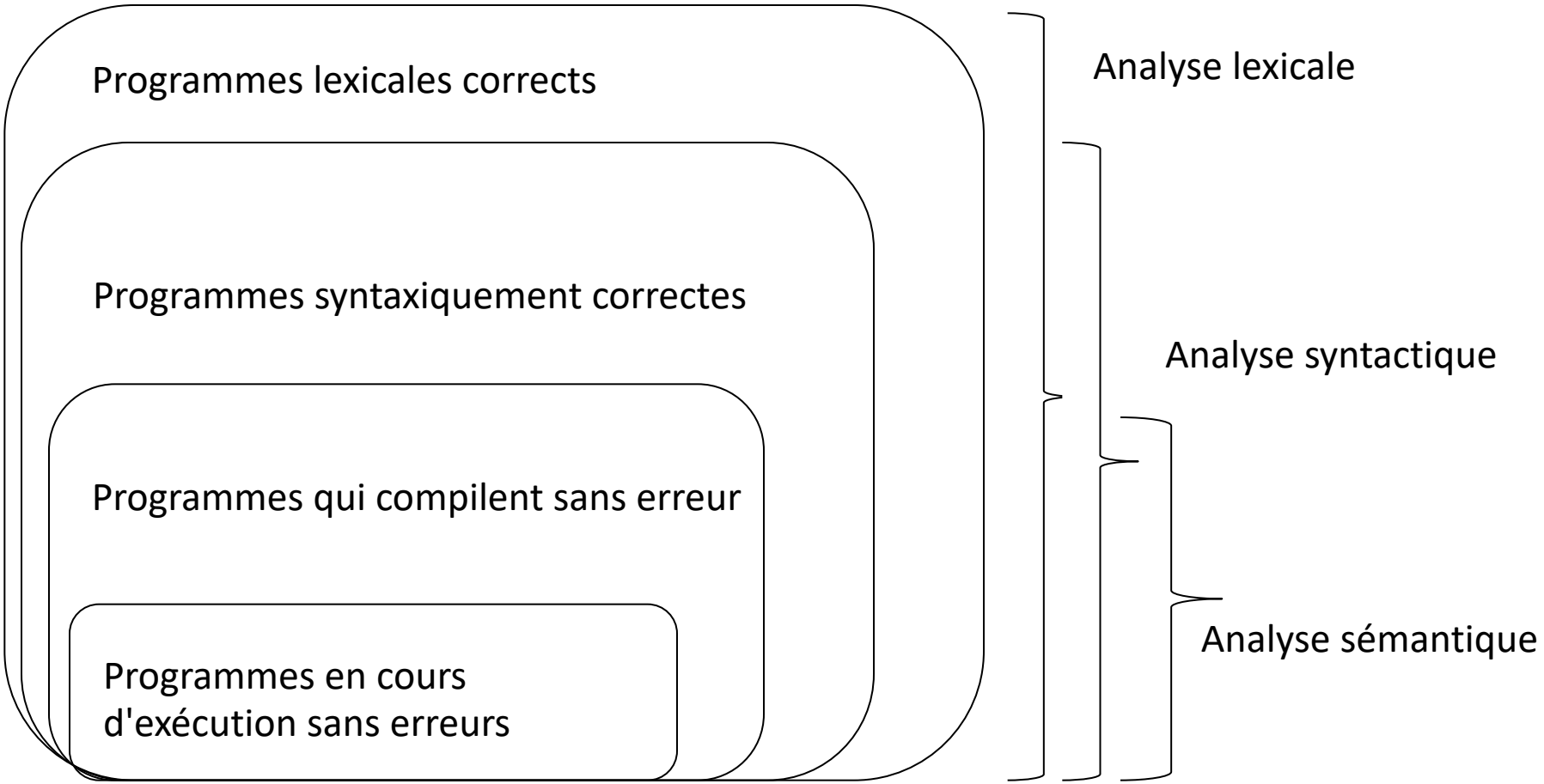
Bogdan Nitulescu

Frontend

ALF
language crunch



Analyse sémantique



Analyse sémantique

ALF
language crunch

- Nous sommes intéressés par
  - Annotez l'arbre de syntaxe avec des informations de type
  - Créer le tableau des symboles
  - Ajouter des noeuds "typecast"

- La plupart de l'analyse sémantique se réfère à la gestion du contexte

Analyse sémantique

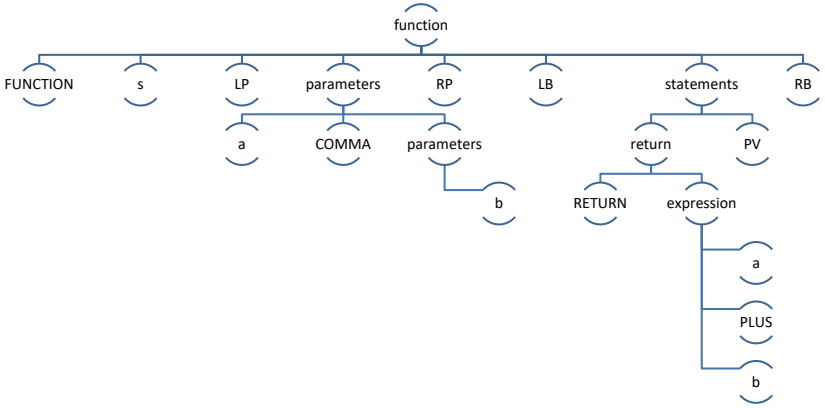
ALF
language crunch

- a=b;
- e = a*b;
- sum (a, b);
- a[i] = s;
- s.element = 7;

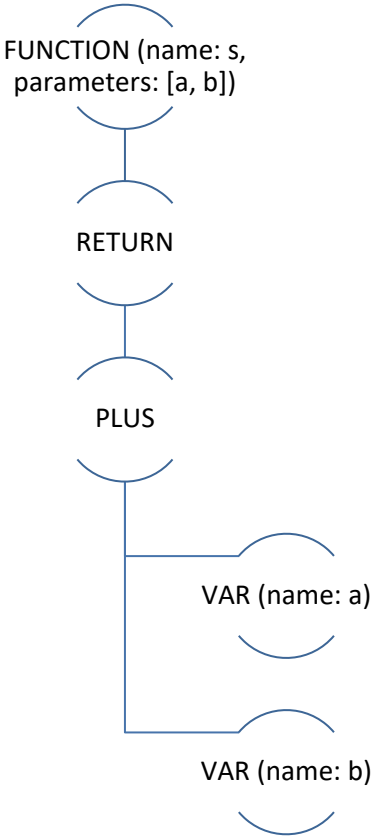
Arbre de syntaxe abstraite

ALF
language crunch

Parse Tree



AST



Exemple de AST

ALF
language crunch

```
function factorial (n)
{
        var f = 1;
        for (var i=1; i < n; i++)
                f = f * i;
        return f;
}
```

Exemple de AST

ALF
language crunch

```
{
        “type”: “function”,
        “name”: “factorial”,
        “parameters”: {
                “n”: “var”
        },
        statements: [ … ]
}
```

Contexte (scope)

ALF
language crunch

- Les contextes mémorisent les déclarations
  - Nom et structure de type
  - Nom de variable
  - Nom, type de retour et les paramètres pour les fonctions
- Lorsqu'elles sont déclarées, les variables, les types et les fonctions sont ajoutés au contexte
- Lorsqu'ils sont consultés, ils sont recherchés dans le contexte actuel
- Les contextes sont imbriqués

Contexte

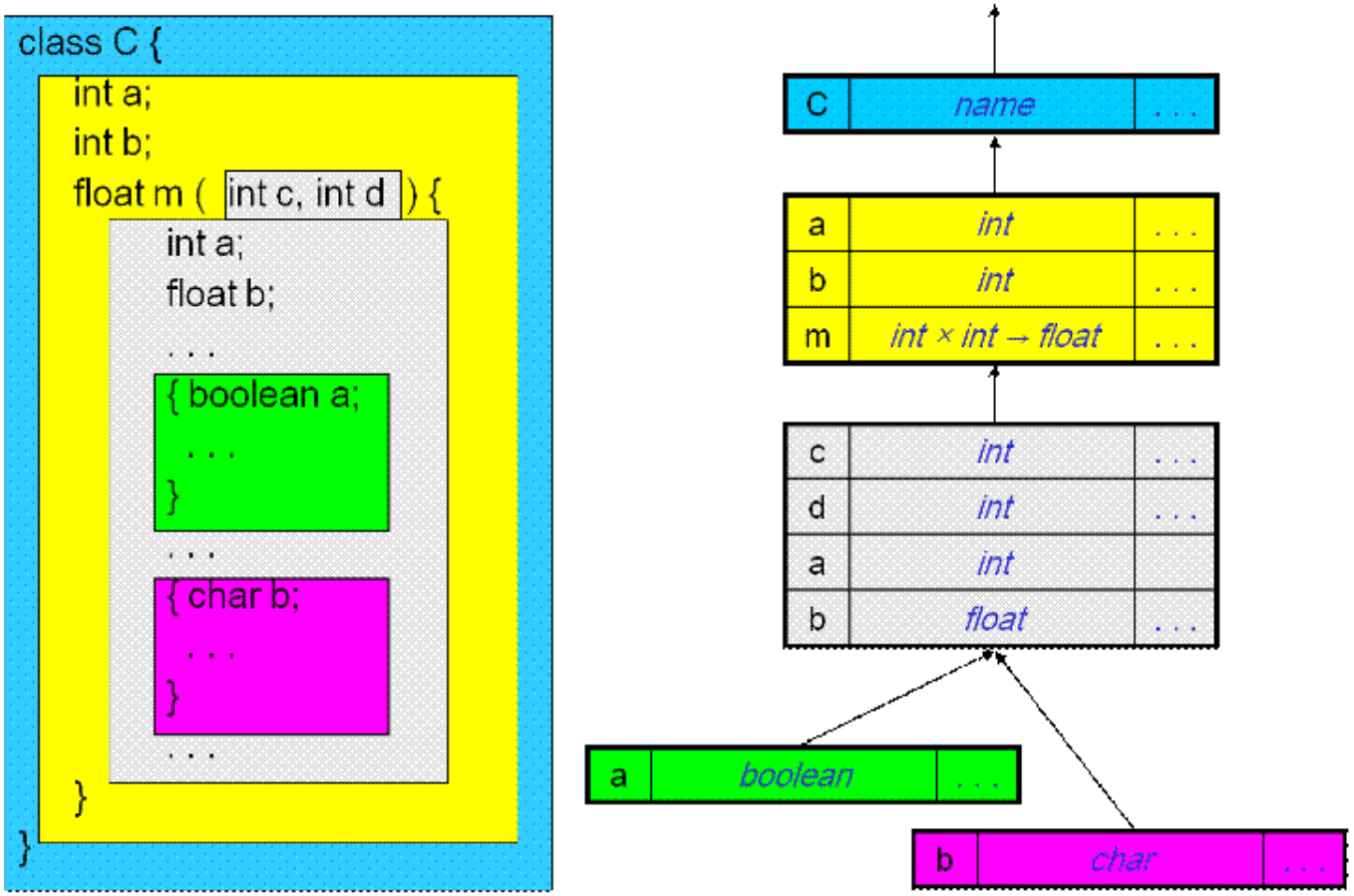


Image from http://staff.polito.it/silvano.rivoira/HowToWriteYourOwnCompiler_file/Page392.htm

Contexte - exemple

ALF
language crunch

- C++
  - Locale (block { … } ou fichier)
  - Label - le contexte c'est la fonction
  - Attributs/méthodes – toute la classe.

- Java
  - Niveaux: Package, Class, Inner class, Method

Contexte et espace de nom

ALF
language crunch

- C:
  - typedef int foo; foo foo;
  - int int;
- Java
  - Integer Integer = new Integer(4);
- C, Java:
  - int foo(x) { return x+4;}
  - int f() { int foo=3; return foo(foo);}

Implémentation du contexte

ALF
language crunch

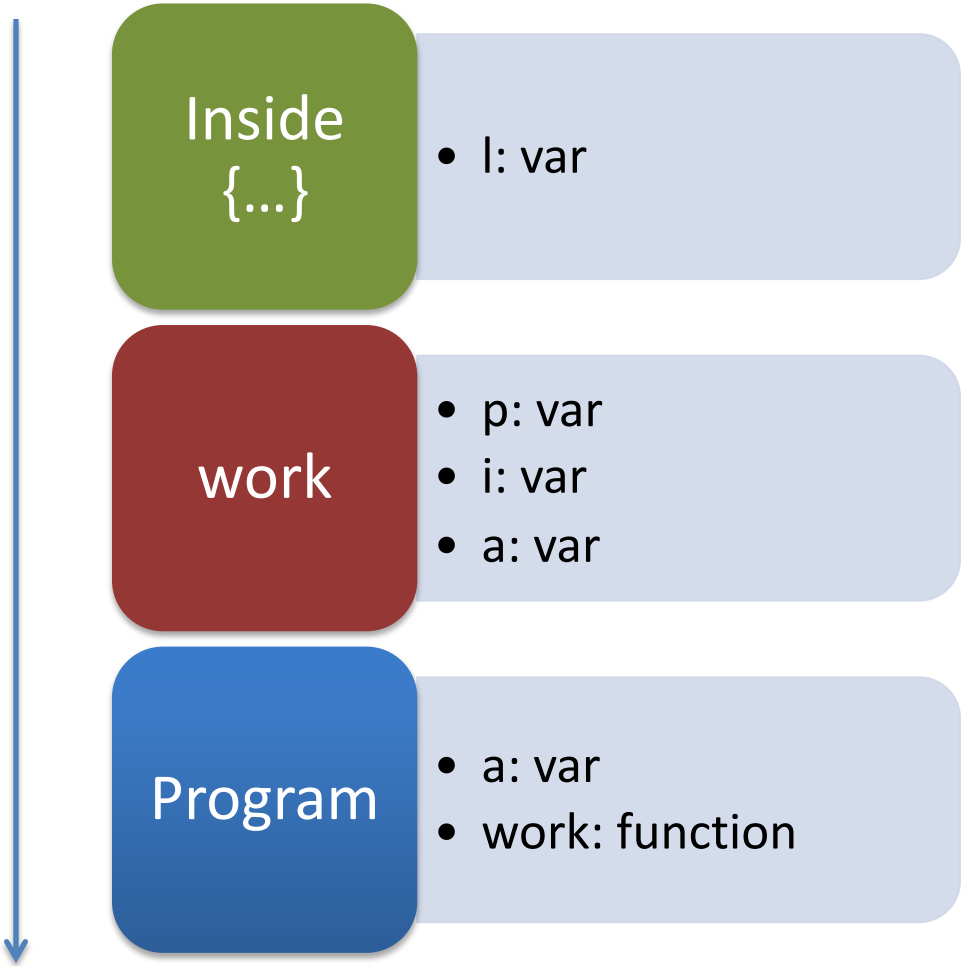
- Tableau de symbole
- Actions:
  - **Nouveau** contexte
  - **Ajoute** un symbole
  - **Retrouve** un symbole
  - **Ferme** le contexte
- Pile ou hashtable

Pile

ALF
language crunch

```
var a;

function work (p)
{
        var i;
        {
                var a;
                let l;
        }
}
```



Hashtable

ALF
language crunch

```
var a;

function work (p)
{
        var i;
        {
                var a;
                let l;
        }
}
```

Symbol	Type
a_program	var
work_program	function
i_work	var
a_work	var
l_work_inner	var

Exemple de AST

ALF
language crunch

```
function factorial (n) // make new context (function)
{
        // add variable n to conext
        var f = 1; // add variable f to conext
        for (var i=1; i < n; i++) // make new context (for),
                                  //add variable i to context (function)
                f = f * i;
                // destroy context (for)
        return f;
        // destroy context (function)
}
```

Contexte statique et dynamique

- Statique - a la compilation
  - C/C++
  - Java
  - Pascal
- Dynamique – lors de l'exécution
  - Javascript
  - Python
  - Ruby

Types

ALF
language crunch

- Type
  - Valeurs autorisées
  - Opérations autorisées

- Types
  - Simple
    - int, float, double, char, bool
  - Composée
    - array, string, pointer, struct
  - Complexe
    - listes ,arbres

Paramètres pour les type

ALF
language crunch

- Simple
  - Le type (int, char, float …)
- Compose
  - struct
    - Liste avec les components
      - Nom
      - Type
  - array
    - Type des éléments
    - Numéro des éléments
    - Les index

Les types sont pour

ALF
language crunch

- Constantes
- Variables
- Fonctions
- Expressions
- Instructions

Langage et type

ALF
language crunch

- **Dynamique** vs. **Statique**
  - Ou est-ce que la vérification de type est effectuée?
    Exécution vs compilation
- **Strongly typed** vs. **Weakly typed**
  - Que se passe-t-il si les types ne correspondent pas?
    Erreur vs. conversion

Vérification de type

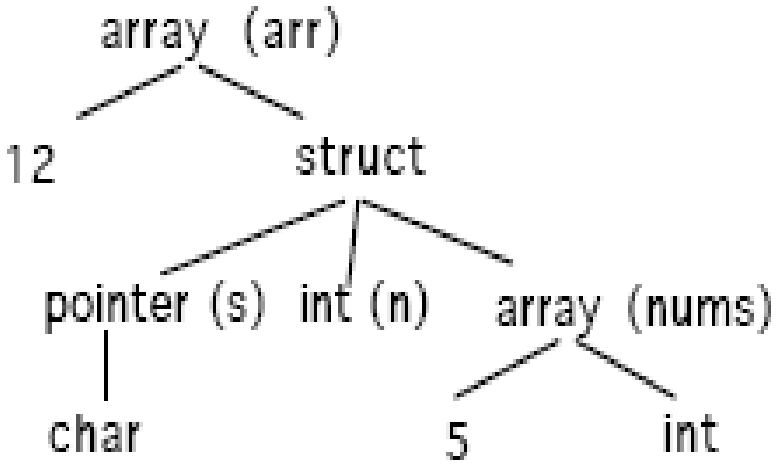
ALF
language crunch

- Synthèse
  - Détermination de type pour un instruction
    - Expression
    - Overloading
- Inférence
  - Détermine un type dans le contexte

Equivalence de types compose

- Un arbre pour l'information de type
  - Nom
  - Structure
- Vérifiez de façon récursive que l'arbre correspond

```
struct {
    char *s;
    int n;
    int nums[5];
} arr[12];
```



Inférence de type

ALF
language crunch

- Déduction de type pour un expression dans le contexte
- A compilation ou a exécution.
- C’est important
  - Vérification de type
  - Overloading de fonction
  - Conversion implicite
    - Widening / Narrowing

Sujets

ALF
language crunch

- Types
- Arbre de syntaxe abstraite
- Analyse sémantique

Questions

ALF
language crunch