Notions abordées dans les vidéos

28 février 2022

Comme demandé par plusieurs étudiants, le présent document vise à lister les principaux mots-clefs et notions abordées dans les vidéos. Cette liste est non-exhaustive, et mise à jour au fur et a mesure de la production des vidéos.

- **01-course-overview**
  - Présentation du contexte du cours
  - Modes d'évaluations
  - Références utiles
- **02-what-is-a-compiler**
  - Présentation des outils composant un compilateur : préprocesseur, compilateur, assembler, et linker
  - Notion de cross-compiler
  - Explication du bootstraping
- **03-compiler-architecture**
  - Différences entre Front-end, Middle-end, et Back-end
  - Détails des différentes partie d'un compilateur : analyseur lexical, analyseur syntaxique, binder, type checker, traduction en code intermédiaire, Canonisation, sélection des instruction, analyse de vivacité et allocation des registres.
  - Vue d'ensemble de GCC et de Clang
- **04-compilation-strategies**
  - Compilation AOT et JIT
  - Notion de bytecode
  - Interpréteurs et machines virtuelles
- **05-tiger-language**
  - Syntaxe, sémantique, et inspiration pour le langage
  - Les choses interdites (Nil, combinaison invalides...)
  - Extensions pour les objets et imports
- **06-tigrou**
  - Étape 1 : scanner, parser et AST
  - Étape 2 : Binder et Type-checker

  - Étape 3 : Génération de code C

- **07-tiger**
  - Historique, contexte et motivations du projet
  - Règles du jeu et références utiles
  - Précisions sur l'architecturation du projet
- **08-dev-tools**
  - Liste des outils a connaître pour gérer Tiger/Spider
  - Rappels sur les autotools
  - Exemples de configure.ac et Makefile.am
- **09-lexical-analysis**
  - Fonctionnement d'un analyseur lexical
  - Notions d'automates, automates déterministes, d'expressions rationnelles et de tokens
  - Notion de matrice de transition
- **10-flex**
  - Description de Flex, un générateur d'analyseur lexical
  - Fonctionnement et fichiers types
  - Simuler la commande wc avec flex
- **11-syntactic-analysis**
  - Notions de Gammaires (LR, LL, GLR, LALR)
  - Dérivations et arbres de dérivation
  - Parseurs prédictifs
- **12-bison**
  - Description de Bison, un générateur d'analyseur lexical
  - Fonctionnement et fichiers types
- **13-coupling-flex-bison**
  - Description des interactions entre Flex et Bison
  - Tracer les erreurs avec les locations
  - Notions de Variants
- **14-error-recovery**
  - Notion de reprise sur erreur dans un parseur
  - Notion de reprise sur erreur locale
  - Notions de reprise sur erreur globale (Burke-Fisher)
- **15-ast**
  - Différence entre AST et Arbre de dérivation
  - Notion de grammaire abstraite et concrète
  - Bonne modélisation objet
- **16-pretty-printer**
  - Comment tester les premiers modules d'un compilateur ?
  - Notion de coding style

  - Utilisation du dispatch dynamique

- **17-efficient traversal**
  - Surcharge et extension d'une hierarchie de classe
  - Clarifications sur les multiméthodes
  - Description des problèmes liés au dispatch dynamique

- **18-visitors**
  - Design pattern visitor et composite
  - Utilité des visitors dans un compilateur
  - Méthodes accept et visit

- **19-further-with-visitors**
  - Notion de combinator pour spécifier des stratégies de parcours
  - Fonctors
  - Default visitors

- **20-syntactic-sugar**
  - Notion de sucre syntaxique
  - Tweast (désucrage en grammaire concrète)
  - Mise en garde lors du désucrage

- **21-tiger-ast-visitors**
  - Hierarchie de classe et spécificités du compilateur Tiger
  - Listing des passes dans TC