

FIIL PIA Project :

Equivalence entre Formules propositionnelles et arbres de décision binaires (BDT's) pour un simple prouveur automatique

October 24, 2018

Burkhart Wolff, Master FIIL Université Paris Saclay

Abstract

Ce projet vise la modélisation et le développement d'un simple prouveur automatique sur la logique propositionnelle à base des BDD (Binary decision trees). Le développement reste sur un niveau abstrait, mais exécutable.

Modelization

Le skeleton de cette théorie consiste des composantes suivantes:

1. `BDT.thy` contient la théorie des BDD's, alors leurs définition d'AST's sous forme d'un type inductive et une *sémantique dénotationnelle*, une fonction $I_{bdd}[[t]]\gamma$ that assigns to each regular expression r the "language" it denotes: $L(r)$.
2. `Formula.thy` définit le type inductive des formules propositionnelles et une compilation d'eux en terme de BDD's, et
3. `PLProver.thy` contient un peu de théorie sur le prouveur construit.

Formules de la Logique Propositionnelle

Les éléments clés de la modélisation sont:

1. représentation des variables : `nat`,
2. représentation des constantes propositionnelles \top , \perp par un booléen,
3. un type pour les connecteurs binaires, et
4. un type inductive pour la syntaxe abstraite des formules.

```

type_synonym atom = bool

datatype binop = And | Or | Impl

(* inductive definition of abstract syntax trees for formula *)

datatype form =
  Var "var"
| Atom "atom"
| Neg "form"
| Bin "binop" "form" "form"

```

La sémantique dénotationnelle est donc défini comme suivant:

```

fun interp ("Iform") where
  "Iform (Var x) I = I x"
| "Iform (Atom a) I = a"
| "Iform (Neg Q) I = (¬ (Iform Q I))"
| "Iform (Bin And Q R) I = ((Iform Q I) ∧ (Iform R I))"
| "Iform (Bin Or Q R) I = ((Iform Q I) ∨ (Iform R I))"
| "Iform (Bin Impl Q R) I = ((Iform Q I) → (Iform R I))"

```

Binary Decision Trees

Les éléments clés de la modélisation sont:

1. Représentation arborescente des formules proche de la sémantique
2. Définition $t := \top | \perp | IF(x, t, t)$
3. Version ordonnée : les variables sont ordonnés, croissantes sur les branches
4. Version réduite : deux sous-arbres différents (non traité) Sémantique
5. Opérations de construction
6. Preuves de correction
7. Fichier BDT.thy

Définitions:

```

fun "interpret":: "bdt ⇒ interpretation ⇒ bool" ("Ibdt [_] _")
where "Ibdt[Atom a] γ = a"
      | "Ibdt[IF x P Q] γ = (if γ x then Ibdt[P]γ else Ibdt[Q]γ)"

```

Propriétés de base:

1. soit $f \in BDT \Rightarrow BDT$ stable par équivalence ($A \equiv B \rightarrow f(A) = f(B)$)
2. on a $f(if(x, t, u)) = if(x, f(t), f(u))$
3. il suffit donc de se donner $f(\top)$ et $f(\perp)$.

On pourrait faire de même pour la conjonction ...

Une définition générale — qui respecte aussi l'ordre important pour la suite — sera comme suivant pour $f(t, u)$:

- $f(a_1, a_2)$ si a_1, a_2 atomique
- $f(a, IF(x, l, r)) = IF(x, f(a, l), f(a, r))$ si a atomique
- $f(IF(x, l, r), a) = IF(x, f(l, a), f(r, a))$ si a atomique
- $f(IF(x, l_1, r_1), IF(x, l_2, r_2)) = IF(x, f(l_1, l_2), f(r_1, r_2))$
- $f(IF(x_1, l_1, r_1), IF(x_2, -, -)) = IF(x_1, f(l_1, u), f(r_1, u))$ si $x_1 < x_2$
- $f(IF(x_1, -, -), IF(x_2, l_2, r_2)) = IF(x_2, f(l_2, u), f(r_2, u))$ si $x_1 > x_2$

Croissance des variables dans les BDT

1. Plusieurs possibilités:

- Fonction récursive qui teste qu'un bdt est ordonné
- Définition inductive; par exemple cette variante (correcte?):

```

inductive ordered :: "nat ⇒ nat ⇒ bdt ⇒ bool"
where
  Oatom : "ordered n i (Atom a)"
  | Oif   : "i ≤ x ∧ x < n
            ⇒ ordered n (Suc x) l
            ⇒ ordered n (Suc x) r
            ⇒ ordered n i (IF x l r)"

```

2. Les opérations logiques sur les bdt préservent la croissance
3. valeur d'une formule : valeur des variables dans les bornes
4. Définition de l'opération $\exists x.t$ correspondant à $t[x \leftarrow \top] \vee t[x \leftarrow \perp]$

Taches

1. Compléter tous les "trous" dans la modélisation
2. Compléter tous les "trous" dans les preuves
3. A quelle condition sur i, j, n, m a-t-on $ordered\ n\ i\ t \rightarrow ordered\ m\ j\ t$? Énoncer et prouver le lemme.
4. Montrer que $form2bdt$ renvoie un BDT ordonné entre 0 et le successeur de la plus grande variable de la formule.
5. Tester qu'un BDT (ordonné) est une tautologie:
 - écrire une fonction booléenne qui teste si un BDT correspond à une tautologie
 - Énoncer et prouver la correction de ce test, on pourra distinguer le cas où le test est positif et le cas où le test est négatif