Calculabilité

TP12

Y. Deville
C-H. Bertrand Van Ouytsel - V. Coppé - A. Gerniers
N. Golenvaux - M. Parmentier

Mai 2021

Question 1 du test

Question : Vrai ou faux : il est possible de créer un langage de programmation qui (en tant que formalisme de calculabilité) possède la propriété SA (soundness algorithmique) mais pas la propriété SD (soundness des définitions).

Réponse : FAUX. Si la fonction interpréteur $I(n, k)$ de ce langage de programmation est calculable (par une machine de Turing M_I), alors toute fonction ϕ_{n^*} calculable à l'aide de ce langage de programmation est calculable (par une machine de Turing). En effet, il suffit d'utiliser la machine de Turing M_{n^*} telle que pour tout $k \in \mathbb{N}$ on ait $M_{n^*}(k) = M_I(n^*, k)$ (que l'on peut obtenir de façon purement théorique grâce au théorème S-m-n).

Question 2 du test

Question : Le langage BLOOP est un formalisme de calculabilité qui possède les propriétés suivantes :

1. SD, SA, S, non CD, non CA, non U
2. SD, SA, S, non CD, non CA, U
3. SD, non SA, S, non CD, non CA, non U
4. SD, SA, S, CD, non CA, non U
5. SD, non SA, S, non CD, non CA, non U
6. SD, SA, S, CD, non CA, U

Réponse : SD, SA, S, non CD, non CA, non U. Explication : SD et SA car BLOOP est un sous-ensemble du langage Java (qui vérifie SD et SA). U, CD et CA impossible par le théorème d'Hoare-Allison.

Question 3 du test

Question : Vrai ou faux : si un formalisme de calculabilité possède la propriété CA (complétude algorithmique), alors il possède nécessairement la propriété CD (complétude des définitions).

Réponse : VRAI. Supposons CA vrai. Si une fonction f est calculable par une machine de Turing M, la propriété CA nous donne un programme D de notre nouveau formalisme qui calcule cette même fonction. Donc f est calculable à l'aide de notre nouveau formalisme, qui vérifie donc CD.

Question 4 du test

Question : Vrai ou faux : si un formalisme de calculabilité possède les propriétés SD (soundness des définitions) et U (descriptions universelle), alors il possède nécessairement la propriété SA (soundness algorithmique).

Réponse : VRAI. Par U, la fonction interpréteur du nouveau formalisme est calculable à l'aide de ce formalisme. Par SD, toute fonction calculable par ce nouveau formalisme est calculable (par une machine de Turing), donc en particulier l'interpréteur du nouveau formalisme l'est (ce qui correspond précisément à SA).

Question 5 du test

Question : L'affirmation suivante est-elle vraie ? Si un formalisme de calculabilité possède les propriétés SA (soundness algorithmique) et CD (complétude des définitions), alors il possède nécessairement la propriété U (descriptions universelle).

Réponse : VRAI. Par SA, la fonction interpréteur du nouveau formalisme est calculable (à l'aide d'une machine de Turing). Par CD, toute fonction calculable est calculable par ce nouveau formalisme, donc en particulier l'interpréteur du nouveau formalisme l'est (ce qui correspond précisément à U).

Question 6 du test

Question : Vrai ou faux : les automates finis déterministes fournissent un formalisme de calculabilité qui possède la propriété SD (soundness des définitions).

Réponse : VRAI. Toute fonction calculable par un automate fini déterministe peut être calculée par une machine de Turing.

Question 7 du test

Question : Vrai ou faux : les automates finis déterministes fournissent un formalisme de calculabilité qui possède la propriété CD (complétude des définitions).

Réponse : FAUX. Les automates finis ne permettent de calculer que des fonctions totales. Par le théorème d'Hoare-Allison, ils ne permettent donc pas de calculer toutes les fonctions calculables (par une machine de Turing), en particulier leur fonction interpréteur.

Question 8 du test

Question : Vrai ou faux : les automates finis déterministes fournissent un formalisme de calculabilité qui possède la propriété U (description universelle).
Réponse : FAUX. Les automates finis ne permettent de calculer que des focntions totales. Par le théorème d'Hoare-Allison, ils ne permettent donc pas de calculer toutes les fonctions calculables (par une machine de Turing), en particulier leur fonction interpréteur.

Question 9 du test

Question : Vrai ou faux : les automates finis déterministes fournissent un formalisme de calculabilité qui possède la propriété SA (soundness algorithmique).

Réponse : VRAI. Il est tout à fait possible de simuler un automate fini déterministe quelconque en Java/Python ou à l'aide d'une machine de Turing. Autrement dit, la fonction interpréteur des automates finis est calculable.

Question 10 du test

Question : Vrai ou faux : les automates finis déterministes fournissent un formalisme de calculabilité qui possède la propriété CA (complétude algorithmique).

Réponse : FAUX. Puisque ce formalisme ne vérifie pas CD, il est impossible qu'il vérifie CA (puisque CA $\Rightarrow$ CD).

Question 1 du TP

Question : Prove the following implication :
$CA \wedge SD \wedge U \Rightarrow SA \wedge CD \wedge S$

Question 1 du TP

Question : Prove the following implication :
$CA \wedge SD \wedge U \Rightarrow SA \wedge CD \wedge S$
Réponse : Considérons un formalisme de calculabilité D qui vérifie CA, SD et U. Puisque la fonction interpréteur de D est D-calculable et que toute fonction D-calculable est calculable (SD), la fonction interpréteur de D est calculable (SA). Puisqu'il existe un compilateur calculable capable de transformer toute machine de Turing M en un programme P de D qui calcule la même fonction (CA), toute fonction calculable (par une machine de Turing) est calculable à l'aide d'un programme de D (CD).
Reste à démontrer S : soit une fonction $f(x, y)$ qui est D-calculable et son programme associé $P(x, y)$. Par SA, elle est calculable : notons $M(x, y)$ la machine de Turing qui la calcule. La propriété S est vraie pour les machines de Turing : il existe S telle que pour tout $x, y \in \mathbb{N}$, $M(x, y) = [S(x)](y)$. Par CA, il existe un programme S_D de D qui calcule la même fonction, c'est-à-dire tel que pour tout $x, y \in \mathbb{N}$, $P(x, y) = [S_D(x)](y)$. Donc D vérifie S.

Question 2 du TP

Question : What is the difference between U and SA ? Does one imply the other ? If not, what property do you have to add to SA to imply U ?

Question 2 du TP

Question : What is the difference between U and SA ? Does one imply the other ? If not, what property do you have to add to SA to imply U ?

Réponse : aucune des deux propriétés n'implique l'autre (par contre, SA avec CD implique U [1]). Donnons deux contre-exemples. Les automates finis déterministes vérifient SA mais pas U. Le langage Python/Java avec un oracle pour l'ensemble $\{k \in \mathbb{N} \mid \varphi_k(42) = 42\}$ vérifie U mais pas SA.

1. Voir question 5 du test ci-dessus.

Question 3 du TP

Question : Among the properties SD, CD, SA, CA, U, and S, which ones hold for the following formalisms ?

1. Java / Python
2. The Oracle Turing Machines, where the oracle set is $K = \{n | P_n(n) \text{ terminates}\}$ (where P_n is the n-th Turing Machine)
3. The language BLOOP with an oracle-program, where the oracle set is $K = \{n | P_n(n) \text{ terminates}\}$ (where P_n is the n-th BLOOP program)

Question 3 du TP

Question : Among the properties SD, CD, SA, CA, U, and S, which ones hold for the following formalisms ?

1. Java / Python
2. The Oracle Turing Machines, where the oracle set is $K = \{n | P_n(n) \text{ terminates}\}$ (where P_n is the n-th Turing Machine)
3. The language BLOOP with an oracle-program, where the oracle set is $K = \{n | P_n(n) \text{ terminates}\}$ (where P_n is the n-th BLOOP program)

Réponse :

1. SD, CD, SA, CA, U, et S
2. CD, CA, U et S
3. SD, SA et S

Question 4 du TP

Question : True or false ?

1. There exist languages that have the SD property, but not the SA property.
2. A formalism satisfying the SA, SD and U properties is a good computability formalism.

Question 4 du TP

Question : True or false ? There exist languages that have the *SD* property, but not the *SA* property.

Réponse : VRAI. Exemple : l'ensemble des programmes Java/Python avec un bonus des programmes-oracles tels que $P_i(x) = [\text{return halt}(i, 0)]$ (pour tout $i \in \mathbb{N}$, avec halt la fonction halt des programmes Java/Python). Comme pour tout $i \in \mathbb{N}$, φ_{P_i} est constante, elle est calculable. Donc ce formalisme vérifie SD. Mais il ne vérifie pas SA car il est impossible de déterminer de façon calculable quelle sera la valeur de cette constante. Si ce formalisme de calculabilité vérifiait SA, alors cela impliquerait que l'ensemble $\{i \in \mathbb{N} \mid P_i(0) \neq \perp\}$ est récursif, ce qui est faux.

Question 4 du TP

Question : True or false ? A formalism satisfying the SA, SD and U properties is a good computability formalism.

Réponse : FAUX. Contre-exemple : le formalisme qui contient un unique programme : [return 1].

Challenge 1

Question : Prove $CD \wedge S \Rightarrow CA$

Challenge 1

Question : Prove $CD \wedge S \Rightarrow CA$
Réponse : soit D un formalisme de calculabilité qui vérifie CD et S. Si P est un formalisme de calculabilité qui vérifie SA, puisque la fonction interpréteur de P est calculable (SA de P) et que toute fonction calculable est D-calculable (CD de D), la fonction interpréteur de P est D-calculable. Notons D_p l'élément de D qui calcule la fonction interpréteur de P. Puisque P vérifie SA, il vérifie SD. Toute fonction P-calculable est donc calculable (SD de P), et donc D-calculable (par CD de D). Soit le programme d'indice $k \in \mathbb{N}$ du formalisme P. Par ce qui est dit ci-dessus, φ_k est D-calculable. Puisque D vérifie S, il existe un transformateur de programme S calculable pour le programme D_p tel que $\forall n, x \in \mathbb{N} : D_p(n, x) = [S(n)](x)$. En particulier, $\forall x \in \mathbb{N} : D_p(k, x) = [S(k)](x)$. On a donc un programme $S(k)$ (du formalisme D) qui calcule la même fonction que le programme d'indice k de P. Le compilateur calculable recherché est donc S et D vérifie CA.

Challenge 2

Question : Show that there exists langages that have the CD property, but not the CA property.

Challenge 2

Question : Show that there exists langages that have the CD property, but not the CA property.

Réponse : this one is really difficult... Good luck !