Notes de cours de Vérification

Yann Miguel

14 janvier 2022

Table des matières

1 Cours 1

Le μ-calculest une logique modale à points fixes.
Symboles du μ-calcul:

- tt: vrai
- ff: faux
- $\wedge$: conjonction
- $\vee$: disjonction
- $\Box$: universel(successeur toujours vrai)
- $\Diamond$: existentiel(au moins un successeur vrai)
- μX.φ: plus petit point fixe
- νX.φ: plus grand point fixe

L'existentiel ne peux pas être vrai si il n'existe pas de successeur à l'état courrant.

Formules importantes:

- $\Diamond$tt: indique la présence d'au moins un successeur
- $\Box$ff: indique l'abscence de successeur
- $\Diamond$ff: comme ff, toujours faux
- $\Box$tt: comme tt, toujours vrai

$\mu X^0.\varphi = \varphi[\frac{ff}{X}]$. $\mu X^{0+1}.\varphi = \varphi[\frac{\mu X^0.\varphi}{X}]$. $\lfloor\lceil \mu X.\varphi \rceil\rfloor = \bigcup_{i=\mathbb{N}} \mu X^i.\varphi$

$$\mu X^0.a \vee \Diamond X = a \vee \Diamond ff = a$$

$$\mu X^1.a \vee \Diamond X = a \vee \Diamond a$$

$$\mu X^2.a \vee \Diamond X = a \vee \Diamond (av\Diamond a)$$

Donc, μX^i.a$\vee\Diamond$X $\rightarrow$ $\exists$ a dans au plus i pas.
Le plus petit point fixe dit donc qu'il existe un a accessible, sans préciser quand.

$$\mu X^0.a \wedge \Box X = a \wedge \Box ff$$

$$\mu X^1.a \wedge \Box X = a \wedge (a \wedge \Box ff)$$

μX^i.a$\wedge\Box$X: a est vrai jusqu'à la profondeur i.
μX^i.a$\wedge\Box$X=$\Box$a, et tout les chemins sont finis.
$\nu X^0.\varphi = [\frac{tt}{X}]$. $\nu X^1.\varphi = [\frac{\nu X^0 \varphi}{X}]$. $\lfloor\lceil \nu X.\varphi \rceil\rfloor = \bigcap_{i\in\mathbb{N}} \nu X^i.\varphi$

$$\varphi = \nu X; a \wedge \Diamond X$$

$$\varphi^0 = a \wedge \Diamond tt$$

$$\varphi^1 = a \wedge \Diamond(a \wedge \Diamond tt)$$

$$\varphi^2 = a \wedge \Diamond(a \wedge \Diamond(a \wedge \Diamond tt))$$

φ^i:$\exists$ un chemin de longueur i + 1, ayant un successeur, $\wedge\Box$a. J'ai un chemn infini avec des a partout.

$$\varphi = \nu X; a \wedge \Box X$$

$$\varphi^0 = a \wedge tt$$

$$\varphi^1 = a \wedge \Box(a \Box tt)$$

φ^i: a est vrai pendant i pas. Donc, φ: a est toujours vrai. Quelques formules:

- μX.$\Box$X: indique que tout chemin se termine, car la formule finira toujours par $\Box$ff, qui est l'abscence de successeur
- μX.($\Diamond$X)$\vee$(νYa$\wedge$.$\Box$Y):Il existe une partie accessible où a est toujours vrai.
- νY.($\Box$Y)$\wedge$(μX.a$\vee\Diamond$X): Il existe toujours un a accessible.
- νY.($\Box$Y)$\wedge$($\neg$req$\vee$(μX.ack$\vee$($\Box$X$\wedge\Diamond$X))): si on voit un req, il y aura toujours un ack dans le futur.
- νX.μY.$\Diamond$Y$\vee$(a$\wedge\Diamond$X): $\exists$ un chemin avec a infiniment souvent présent.
- μX.νY.($\Box$Y$\wedge\neg$a)$\vee$a$\wedge\Box$X: $\exists$ un nombre fini de a pour toute exécution.

Un jeu de parité réponds à la question: Si j'ai un graphe, est-ce que la formule est vraie dedans?
Un jeu de parité peux créer deux types de cycles avec les points forts, qui sont nommés en fonction du gagnant lorsqu'on entre dans le cycle; Ils sont:

1. cycle μ-perdant
2. cycle ν-gagnant

2 Cours 2

Un jeu de parité est un ensemble G=(V,E,Ω), avec:
- V, des positions
- E, des arrêtes
- Ω, une fonction

Les positions du jeu de parité peuvent apprtenir à un des deux joueurs.
Joueur 1(aussi appelé Eve):
- Possède les positions circulaires
- Gagne si la plus grande priorité vue infiniment souvent est paire

Joueur 2(aussi appelé Adam):
- Possède les position carrées
- Gagne si la plus grande priorité vue infiniment souvent est impaire

Une stratégie σ est une fonction qui regarde l'historique du jeu.
Elle est dite positionnelle si elle ne regarde pas le dit historique.
Dans la partie π, si $\pi_1 \in$V$_2$, alors π_{i+1}=$\sigma(\pi$[0,i]).
Théorème: Si un joueur gagne dans G, il gagne avec une stratégie positionnelle.
Algortihme de résolution des jeu de parité:
Compléxité Θ(n^d):
Une variante des jeux de parité sont les jeux à registres. En plus

Algorithm 1 Algorithme de Lielonka

f_1(G,d)
repeat
  N_d ={v$\in$V| Ω(v)=d}
  G'=G\Att$_1$(G, N_d)
  W_2=f_2(G, d-1)
  G=G\Att$_2$(G,W_2)
until W_2=$\varnothing$
return G

de jouer, à chaque tour, le joueur 1 range la valeur sortie dans un registre. Il existe k registres.
Fonctionnement des registres:
- mise à jour(le joueur 1 met à jour le registre i):
  - j < i: r_j=0.
  - r_i= p(la valeur sortie)

- $j > i$: r_j=max(r_j,p)
- priorité de sortie:
  - 2i si max(r_i,p) est pair
  - 2i+1 sinon

Les conditions de victoire sont les mêmes que pour les jeux de parité.
Lemme: Si le joueur 2 gagne G, il gagne G_k, $\forall$ k. Si le joueur 1 gagne G, $\exists$k tel que le joueur 1 gagne G_k.
Théorème: Joueur 1 gagne G si et seulement si elle gagne $G_{log(n)+1}$.
Par conséquent, J_1 gagne $G_{log(n)+1}$ $\Rightarrow$$J_1$ gagne G.
Induction:J_1 gagne $G_{log(n)+1}$ avec une stratégie telle que, si $r_k \geq$p est paire, alors il n'y a pas de sortie 2k+1.
Cette stratégie ne fonctionne que si il n'y a un componant fortement connexe utilisant tout les registres. Dans ce cas, il faut un registre de plus pour cette stratégie.
Corollaire: Algorithme $2^{\Theta(log^3(n))}$: résoudre $G_{log(n)+1}$ au lieu de G, et le nombre de priorités est Θ(log(n)).
f_1(G,d,p_2,p_1). Un dominion est un ensemble tel que un joueur a une stratégie gagnante à partir du dominion et qui n'en sort pas.
f_1 doit contenir tout les dominions J_1 de taille$\leq$$p_1$, et n'intersecte pas avec les dominions de J_2, de taille $\leq$$p_2$.
f_2(G,d,p_1,p_2) existe aussi.

Algorithm 2 Algorithme

f_1(G,d,p_2,p_1)
if G=$\varnothing$$\vee$p$\leq$1 **then**
  return G
end if
G_1=f_1(G,d,p_2,$\lfloor \frac{p_1}{2} \rfloor$)
Nd={v| Ω(v)=d}
H=G_i\Att_1(Nd,G)
W_2=f_2(H,d-1,p_1,p_2)
G_2=$G_1 i$\Att_2(W_2,G_2)
G_3=f_1(G_2,λ,p_2,$\lfloor \frac{p_1}{2} \rfloor$)
return G_3

3 Cours 3

Les problèmes de synthèse sont une famille de problèmes génériques dont la recherche a été initiée par Church en 1957. Cette famille introduit un champ de recherche variée, qui est partie à la base de la théorie algorithmique de jeux.
Ce sont des problèmes qui sont, en général, difficiles ou insolubles.
On peux donc se damnder quand peux-t-on les résoudre, et avec quelle compléxité.
Soit R $\subseteq$ X$\times$Y, une relation, et soit f: X$\rightarrow$Y, une fonction partielle.

f uniformise R si:
- domaine(f)=domaine(R).
- $\forall$x$\in$domaine(R), (x,f(x))$\in$R.

On peux aussi dire que f est une fonction de choix pour R(nom moins utilisé).
Soit S, un formalisme de relations X$\times$Y.

Soit P, un formalisme de fonctions de X$\rightarrow$Y.

Algorithm 3 Problème de P-S synthèse

Require: Une relation R donnée dans S(R appelée spécification)
Ensure: Produire, pour un algo, une fonction f donnée dans P(f appelée programme), telle que f uniformise R
R désigne la paire (input,output) considérées comme acceptables.
f désigne le programme qui assigne à chaque input un output acceptable.

Pour définir des spécifications, on utilise MSO(logique pour les spécifications) sur A$\times$B.
Si on a u, un mot infini sur A, et v, un mot infini sur B, on note u$\otimes$v, un mot infini sur $(A,B)^\omega$.
$A_a(x)=\bigvee_{b\in B}(a,b)(x)$. $B_b(x)=\bigvee_{a\in A}(a,b)(x)$.
Le formalisme utilisé pour les programmes est un automate fini déterministe et qui, en plus de lire des lettres, va produire des lettres.
Pour les spécification, on utilise la logique MSO, pour les programmes, on utilise les machines de Mealy.
Deux points pour un programme très simple(comme une machine de Mealy):

1. Réactif: chaque entré produit une sortie
2. Mémoire finie

Complxité non-élémentaire: compléxité qui est une exponentielle infinie non terminale.
Complexité tower: complexité non-élémentaire qui possède n exponentielles, n étant le nombre déntrées. Résolution du problème de Church:

Algorithm 4 Problème de synthèse de Church(compléxité tower)
Require: formule MSO qui définit une spécification R
Ensure: une machine de Mealy qui définit f, tel que f uniformise R

1. Passer de la formule logique MSO à l'automate fini.
2. Passer des automates aux jeux.
3. Jeux de Muller.
4. Jeux de parité.

On peux voir une relation $R \subseteq A^\omega \times B^\omega$ comme un jeu infini entre deux joueurs: J_1 et J_2.
J_1 joue des lettres de A, J_2 joue des lettres de B. Ils jouent tour à tour et J_1 commence.
Qui gagne?

1. Quand le mot ne satisfait pas la spécification.
2. Quand le mot satisfait la spécification.

Théorème: Un langage $L \subseteq \Sigma^\omega$ est définissable en MSO si et seulement si il est reconnu par un authomate de Muller/B$\ddot{u}$chi déterministe.
Automate de B$\ddot{u}$chi: $A=(Q,\Sigma,\delta,q_0,F)$:
- Q: ensemble fini d'états
- Σ: alphabet
- $\delta \subseteq Q \times \Sigma \times Q$
- $q_0 \in Q$: état initial
- $F \subseteq Q$ ensemble des états finaux

Automate de Muller déterministe: $A=(Q,\Sigma,\delta,q_0,F)$:
- Q: ensemble fini détats
- Σ: alphabet fini
- δ: $Q \times \Sigma \to Q$
- $q_0 \in Q$
- $F \subseteq P(Q) \leftrightarrow$ Ensemble des parties de Q

On regarde l'ensemble détats qui apparait infiniment souvent, et on se demande si il est acceptant ou non. L'ensemble d'états est acceptant si il respecte la condition de Muller.

1. On transforme une formule φ MSO en un automate de Muller A_φ qui reconnait le même langage sur $A \times B$.

Définition: Un jeu de Muller est un jeu à deux joueurs sur un graphe (fini). L'ensemble des sommets du graphe est partitionné entre les deux joueurs. Le joueur 2 gagne si la suite des sommets visités lors de la partie satisfait une condition de Muller donnée.
Propriété: Le joueur 2 a une stratégie gagnante à mémoire finie sur le jeu de Muller B_φ si et suelement si le problème de synthèse a une solution. **Lemme**: Tout jeu de Muller est équivalent à un jeu de parité.
Lemme: Tout jeu de parité admet une stratégie gagnante positionnelle pour un des deux joueurs. **Proposition**: Tout automate de Muller déterministe est équivalent à un automate de parité déterministe. Automate de parité déterministe: $A=(Q,\Sigma,\delta,q_0,\pi)$:

- Q: ensemble des états finis
- Σ: alphabet fini
- δ: $Q \times \Sigma \rightarrow Q$
- $q_0 \in Q$
- π: $Q \rightarrow \mathbb{N}$
- Un automate de parité est un cas particulier d'un automate de Muller.
- Un automate de B$\dot{}\cdot u$chi est un cas particulier d'un automate de parité:
  - états finaux$\rightarrow$0
  - états initiaux$\rightarrow$1

Lemme: $\forall$ alphabet Σ, $\exists$ un automate déterministe avec un ensemble détats Q totalement ordonné et une fonction $y:Q \rightarrow P(\Sigma)$ ayant la propriété suivante: $\forall$ mot d'entrée, l'ensemble des lettres apparaissant infiniment souvent est obtenu en appliquant y au plus petit état apparaissant infiniment souvent. **Proposition**: Le problème de Church a une solution $\Leftrightarrow$ J2 a une stratégie gagnante pour un jeu de parité à mémoire finie.

Théorème(B$\dot{}\cdot u$chi-Landueben 1969): J2 a une stratégie gagnante $\leftrightarrow$ J2 a une stratégie gagnante positionnelle. Un des deux joueurs à une stratégie gagnante positionnelle. Dans un jeu de parité, les rôles de J1 et de J2 sont symmétriques: si on augmente de 1 toutes les priorités, les rôle ssont échangés.
Définition: Soit X, un ensemble de positions dans un jeu. On définit l'attracteur de X pour le joueur J_i, $i \in$ 1, 2 par:

- X_0=X
- X_{K+1}=$X_K \cup Y_K \cup Z_K$
  - Y_K: toutes les positions où J_i a une transition vers X_K
  - Z_K: toutes les positions où l'adversaire de J_i n'a que des transitions vers X_K.

Cas de base de l'induction: Une seule priorité utilisée. Sans perte de généralité, on suppose que la priorité est 1. Dans ce cas, J2 gagne si J1 est bloqué.
On définit X comme le J2-attracteur de l'ensemble vide. Depuis X, J2 a une stratégie positionnelle ganante. Hors de X, J1 a une stratégie positionnelle gagnante.