Q18 6.5

Encoding Kripke structure — Model Checking CTL with SFDD

Définition formelle des SFDD :

Soit T un ensemble de termes. L'ensemble des SFDD S est défini inductivement :

$\perp \in S$: le terminal rejetant

$\top \in S$: le terminal acceptant

$\langle t, \tau, \sigma \rangle \in S \iff t \in T,\ \tau \in S,\ \sigma \in S$

– noeuds avec le terme t, sous-noeud acceptant τ (take node), sous-noeud rejetant σ (skip node)

Exemple :

Termes : a<b<c<d

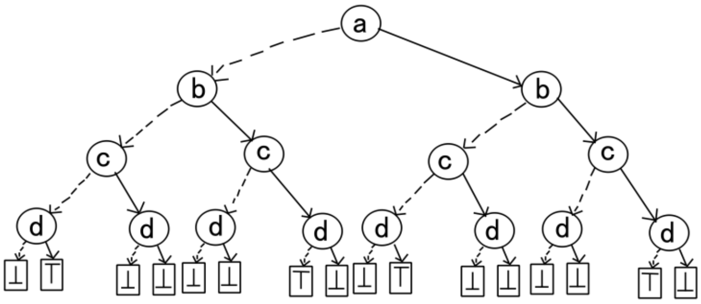
Ensembles :

{a,b,c}

{a,d}

{b,c}

{d}



But : implémenter le CTL model checking dans une structure SFDD

Model checking:

Modèle M, état s -> satisfaction : $M,s \models \phi$ vérifie si la formule ϕ est valide sur s

Les algorithmes de model checking fixent un modèle Krypke $M = \langle S, \rightarrow, AP, \nu \rangle$

et une formule ϕ et calculent $[\phi]_M = \{s \in S \mid M,s \models \phi\}$: dénotation de ϕ

AP = propositions atomiques, v indique quels états vérifient chaque AP

$[\phi]_M$ = l'ensemble d'états satisfaisant la formule

Il faut donc encoder la Kripke structure :

La Kripke structure est donc une structure dans laquelle est décrite le comportement du système

La Kripke structure d'un ensemble de propositions atomiques AP est un tuple $K = \langle S, S_0, R, L \rangle$:

S est un ensemble fini d'états, S_0 est un ensemble d'états initiaux (non vide, $S_0 \subseteq S$)

R est une relation entre les états : $R \subseteq S \times S$ relation binaire "left-total" sur S représentants les transitions

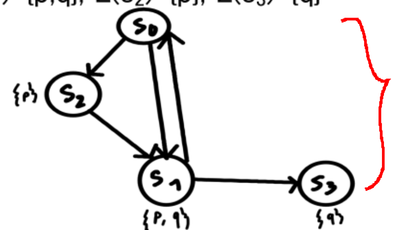
left-total : il y a toujours un successeur à chaque état.

L : $S \rightarrow P(AP)$ labélise chaque état en donnant un ensemble d'AP vérifiés par cet état.

Exemple : $S=\{s_0,s_1,s_2,s_3\}$, $S_0=\{s_0\}$, $R=\{(s_0,s_1),(s_1,s_0),(s_0,s_2),(s_2,s_1),(s_1,s_3)\}$, $L(s_0)=\varnothing$, $L(s_1)=\{p,q\}$, $L(s_2)=\{p\}$, $L(s_3)=\{q\}$

Labelling function : la plus importante (dis quels AP sont vérifiés)

Elle est injective ($\forall k \exists$ un unique s tq $L(s)=k$)



Comment encoder la relation passant d'un état à un autre?

Si on passe d'un état {p,q} à {p',q'}, on dénote la relation (la fllèche) par (p,q,p',q')

Définition du SFDD à partir de la Kripke structure :

Nous créons un ensemble "sibling" de AP : AP' disjoint : $AP \cap AP' = \varnothing$ et une fonction bijective sib:AP->AP'

Dans un SFDD nous avons besoin d'un ordre sur les éléments inclus dans l'ensemble (ici les AP, <)

Nous créons un ordre sur $AP \cup AP'$: <' créé à partir de <

entrelacement des éléments de AP et AP' :

$\forall s_a, s_b \in AP : s_a < s_b \Rightarrow s_a <' sib(s_a) <' s_b$ et $sib(s_a) <' sib(s_b) \Rightarrow sib(s_a) <' s_b <' sib(s_b)$

Donc $s_a < s_b \iff s_a <' s_b \iff sib(s_a) <' sib(s_b)$

Exemple : AP={p,q}, AP'={p',q'}, sib(p)=p', sib(q)=q' et p<q alors p<'p'<'q<'q'

Exemple : encoder une Kripke structure pour SFDD

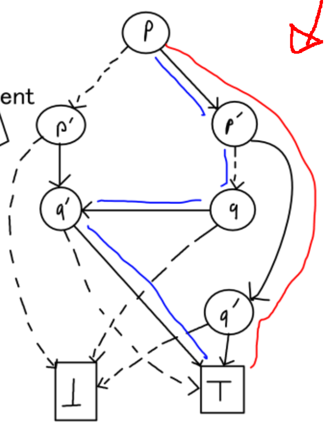
encodage de l'exemple précédent

Soit une Kripke structe $K = \langle S, S_0, R, L \rangle$ Le SFDD est alors :

$G_k = \bigcup_{(s_a,s_b)\in R} enc_{AP \cup AP'}(\{L(s_a) \cup sib(L(s_b))\})$

l'union de tous les encodages des relations sur l'ensemble $AP \cup AP'$

-> encodage de la Kripke structure arc par arc



$s_2 \rightarrow s_1$: p,p',q'

$s_1 \rightarrow s_3$: p,q,q'

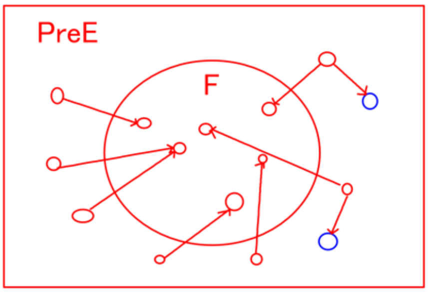
Algorithmes pour les formules CTL.

Nous savons :

AXϕ <=> notEX(notϕ), AFϕ <=>notEG(notϕ), AGϕ <=>notEF(notϕ)

EFϕ <=>E[trueUϕ] (true EU ϕ), A[ϕUθ] <=> notE[notθU(notϕ AND notθ)AND notEG(notθ)]

-> Besoin que des algorithmes pour les opérations EX, EU, EG



EXϕ : Afin de trouver l'ensemble s des états satisfaisants EXϕ :

Soit F l'ensemble des états satisfaisants ϕ.

-> s = preE(F) prédessesseurs existentiel des états de F

-> comment calculer le prédécesseur existentiel?

PreE(F)=reduce$_{AP \cup AP'}$(G_K∩(enc(P(AP))⊗$\overbrace{\text{sib(F)}}^{F'}$),AP) AP

GK : relation entre des éléments de AP et AP' encodage de la F relation entre AP et les états satisfaisant la formule dans F'

reduce : garder que la partie AP et pas AP'

E(ϕUntilΨ) But : trouver l'ensemble S des états satisfaisants E(ϕUntilΨ)

Soit F,G les ensembles d'états satisfaisants ϕ et Ψ; et N un ensemble d'états

S<-G (états satisfaisants Ψ)

N<-enc(∅)

while N!=S do: (vérification de point fixe)

N<-S

S<-S∪(F∩preE(S)) (on ajoute les prédécesseurs de S étants dans F jusqu'à ce qu'il n'y ait rien à ajouter

EG(ϕ) But : trouver l'ensemble S des états satisfaisants EG(ϕ)

Soit F l'ensemble d'états satisfaisant ϕ et N un ensemble d'états

S<-F (états satisfaisants ϕ)

N<-enc(∅)

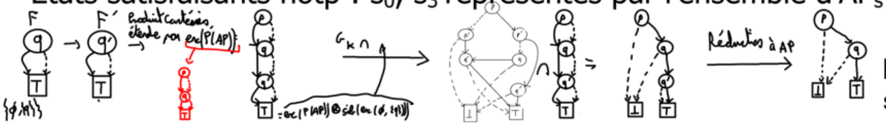
while N!=S do: (vérification de point fixe)

N<-S

S<-S∩preE(S) (on ne garde dans S que ceux qui ont un prédécesseur dans S)

Exemple : Vérifier EX(notp) sur l'exemple ci-dessus

Etats satisfaisants notp : s_0, s_3 représentés par l'ensemble d'AP$_s$: {∅, {q}}



Donc {p,q} est le seul état satisfaisant EX(notp)