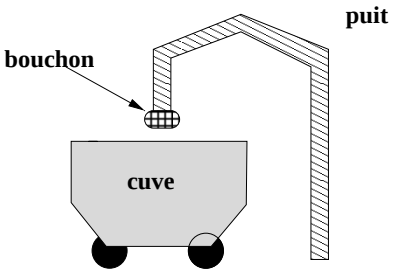
Master 1 d'Informatique. ***Langages de spécification de logiciel***

Gestion de Puits-Cuves

Considérons le problème suivant d'exploitation de fuite de pétrole de deux puits en évitant une pollution.



Chaque puits peut être bouché et cette opération prend un temps $\alpha > 0$. L'enlèvement du bouchon prend le même temps. Pendant ces deux opérations le pétrole peut continuer à couler. Le remplacement d'une cuve en fonction par une cuve vide prend un temps $\beta > 0$. Pour des raisons économiques et techniques il faut remplir chaque cuve avec une quantité maximale de pétrole sans le laisser déborder (c'est un requis). On suppose que le volume libre d'une cuve vide est $> 2\alpha\eta$.

Comme capteurs d'entrée on a un capteur *vLibre* qui indique le volume libre d'une cuve. Lorsque la cuve est absente pour un puits, la valeur de *vLibre* est $undef$. On sait que le flot maximal de pétrole ne dépasse pas η, donc pendant une durée T le volume de pétrole sorti d'un puits est majoré par $\eta \cdot T$.

Le contrôleur doit définir les sorties : *poserBouchon*, *déboucher* et *changerCuve* à valeurs booléennes. *poserBouchon* et *déboucher* correspondent respectivement aux opérations de pose et d'enlèvement du bouchon ; *changerCuve* veut dire que l'opération de changement de cuve est en cours.

Au début les puits sont débouchés et une cuve vide est installée à chaque puits pour y recueillir le pétrole.

a. Donner un vocabulaire qui permet de spécifier un contrôleur.

b. Écrire une ASM pour le contrôleur. Paramétriser les puits pour rendre la machine compacte. Vérifier que les gardes de cette machine sont vraies en des points isolés. À quoi sert la contrainte sur le volume d'une cuve vide ?

c. Considérons la propriété suivante (t est une variable pour le temps et x est une variable pour les puits) :

$$\forall\, t \forall\, x \ \big[\ changerCuve^{\circ}(t,x) \ \rightarrow$$

$$\exists\, t' < t \ \big(\ \forall\, \tau \in (t', t]\, changerCuve^{\circ}(\tau, x) \ \wedge\ \neg\, changerCuve^{\circ}(t', x) \ \wedge\ vLibre^{\circ}(t', x) \leq \eta \cdot \alpha) \ \big) \ \big] \,. \quad (1)$$

Cette propriété, représente-elle la vivacité ? (La vivacité décrit une disponibilité, souvent une disponibilité optimisée).

d. Décrire une propriété de *sûreté*.

Réponses.

a. Donner des vocabulaires qui permettent de spécifier un contrôleur et les requis.

Réponse. Ci-dessous nous donnons 2 versions de contrôleur et discutons leurs propriétés.

Un vocabulaire pour spécifier nos contrôleurs est le suivant.

Sortes : $Puits = \{0, 1\}$ et les sortes par défaut (en particulier le temps $\mathbb{T}$).

Fonctions :
• Entrées : $vLibre : Puits \to \mathbb{R} \cup \{undef\}$, $CT :\to \mathbb{T}$;
• Sorties : $poserBouchon$, $déboucher$ et $changerCuve$ du type $Puits \to Bool$;
• Fonctions internes auxiliaires :
pour Contrôleur 1 : $dl1$, $dl2$, $dl3$ du type $Puits \to \mathbb{T} \cup \{undef\}$,
pour Contrôleur 2 : dl du type $Puits \to \mathbb{T}$;
• α, β, η du type $\to \mathbb{R}$ (fonctions statiques externes, i. e. constantes).

Vocabulaire pour spécifier les requis contient les mêmes sortes, les constantes, les fonctions d'entrée/sortie temporisées. Nous ne devons pas utiliser pas les fonctions auxiliaires dans les requis.

b. Écrire une ASM pour le contrôleur. Paramétriser les puits pour rendre la machine compacte. Vérifier que les gardes de cette machine sont vraies en des points isolés.

Réponse.(Voici 2 solutions,l'une plus simple mais moins robuste, et l'autre un peu plus longue mais plus robuste.)

On suppose que les relations de la forme $x\omega y$, où $x, y \in (\mathbb{R} \cup undef)$ et $\omega \in \{=, <, \leq, >, \geq\}$, sont fausses si l'un argument est dans $\mathbb{R}$ et l'autre est $undef$.

Initialisation : $vLibre(x) > \alpha \cdot \eta$ (c'est un ensemble de valeurs), $\neg\, poserBouchon(x)$ (cette notation dit que la valeur de $poserBouchon(x)$ est **false**), $\neg\, déboucher(x)$, $\neg\, changeCuve(x)$, pour Contrôleur 1 : $dl1(x) = dl2(x) = dl3(x) = undef$,
pour Contrôleur 2 : $dl(x) = undef$.

Contrôleur 1 :

forall $x \in Puits$ **do**
if $vLibre(x) \leq \alpha \cdot \eta \wedge \neg\, poserBouchon(x)$
then $[\, poserBouchon(x) := true,\ dl1(x) := CT + \alpha \,]$
if $CT = dl1(x)$
then $[\, poserBouchon(x) := false,\ changerCuve(x) := true,\ dl2(x) := CT + \beta \,]$
if $CT = dl2(x)$
then $[\, changerCuve(x) := false,\ déboucher(x) := true,\ dl3(x) := CT + \alpha \,]$
if $CT = dl3(x)$
then $déboucher(x) := false$

Le Contrôleur 1 n'est pas implémentable à cause des conditions $CT = dlJ(x),\ \ J = 1, 2, 3$. On peut ne pas les replacer par $CT \geq dlJ(x)$ car cela ferait l'algorithme incorrect (ces dernières conditions restent vraies après l'exécution des parties **then** et on obtient des updates inconsistants). On peut utiliser certaines fonctions internes comme drapeaux pour corriger l'algorithme. Dans ce cas on peut se limiter à une seule fonction dl – voir ci-dessous. Par ailleurs, cet algorithme a un autre défaut : il ne détecte pas assez précisément l'arrivée d'un cuve vide et donc n'optimise pas l'exploitation des puits car il ne prend pas en compte qu'une cuve vide peut arriver avant le temps limite β. Ce défaut est remédié dans

le Contrôleur 2. On voit que ce dernier n'utilise pas β.

Contrôleur 2 :

forall $x \in Puits$ **do**
if $vLibre(x) \leq \alpha \cdot \eta \wedge \neg poserBouchon(x)$
then $[\, poserBouchon(x) := true,\ dl(x) := CT + \alpha \,]$
if $poserBouchon(x) \wedge CT \geq dl(x)$
then $[\, poserBouchon(x) := false,\ changerCuve(x) := true \,]$
if $changerCuve(x) \wedge vLibre(x) \geq 2\alpha \cdot \eta$
then $[\, changerCuve(x) := false,\ déboucher(x) := true,\ dl(x) := CT + \alpha \,]$
if $déboucher(x) \wedge CT \geq dl(x)$
then $déboucher(x) := false$

Dans le Contrôleur 2 nous utilisons le requis que le volume d'une cuve vide est au moins $2\alpha \cdot \eta$. Remarquons que le temps $2\alpha \cdot \eta$ est suffisant pour déboucher et boucher une cuve sans débordement de celle-là.

c. Considérons la propriété suivante (t est une variable pour le temps et x est une variable pour les puits) :

$$\forall\, t\, \forall\, x\ \Big[\ changerCuve^{\circ}(t,x)\ \rightarrow$$

$$\exists\, t' < t\ \Big(\ \forall\, \tau \in (t', t]\, changerCuve^{\circ}(\tau, x)\ \wedge\ \neg\, changerCuve^{\circ}(t', x)\ \wedge\ vLibre^{\circ}(t', x) \leq \eta \cdot \alpha)\ \Big)\ \Big] \quad (1)$$

Cette propriété, représente-elle la vivacité ?

Réponse.

Cette formule représente une partie de la vivacité, à savoir la partie qui dit que le remplissage d'une cuve doit être maximale. Une autre partie de la vivacité dit qu'un puits doit rester bouché le plus courts temps possible, plus précisément, si pour 2 moments $t_1 < t_2$ on a $vLibre = undef$ (une cuve n'est pas installée) sur $(t_1, t_2]$ et $vLibre \neq undef$ au moment t_2 alors on a *déboucher* et $\neg\, poserBouchon$ sur $(t_2, t_2 + \alpha]$.

d. Écrire une propriété de *sûreté*.

Réponse.

Sûreté :

$$\forall\, t\, \forall\, x\ \Big(\ déboucher^{\circ}(t,x)\ \rightarrow\ vLibre^{\circ}(t,x) > 0\ \Big).$$

Rappel : $vLibre^{\circ}(t,x) > 0$ dit que $vLibre^{\circ}(t,x) \neq undef$ et $vLibre^{\circ}(t,x) > 0$.

Une autre part de la propriété de sûreté : si $vLibre \leq \alpha \cdot \eta$ (et donc $vLibre \neq undef$) alors $poserBouchon$ doit être **true**.