Complexité, mais dur
Matthieu Boyer
Planche A

Khôlle MP2I
GRRR

Filtrage par Motif en OCaml

On s'intéresse à la construction `match ... with ...` d'OCaml. Ici, les constructeurs des types algébriques sont notés $C\,(e_1, \dots, e_k)$ où k est l'arité du constructeur C. On rappelle la définition des listes :

```
type 'a list = Empty | Cons of 'a * 'a list
```

Dans la suite on considère que toutes les valeurs OCmal sont des constructeurs $v := C\,(v_1, \dots, v_k)$. En particulier, les constantes `true` et `false` sont des valeurs du type `bool = true | false`. Les motifs sont où :

- Des identifiants de variables/
- Le motif joker `_`.
- Un constructeur appliqué à k motifs.
- La disjonction de deux motifs.

On introduit une notion de matrice de filtrage, selon la transformation illustrée ci-dessous. Les expressions à droite des motifs (a_i) sont appelées des actions :

$$\begin{array}{l}\texttt{match (e}_1\texttt{, ..., e}_m\texttt{) with}\\ \texttt{| m}_{1,1}\texttt{, ..., m}_{1,m}\texttt{) -> a}_1\\ \texttt{| ...}\\ \texttt{| m}_{n,1}\texttt{, ..., m}_{n,m}\texttt{) -> a}_n\end{array} \longmapsto \begin{pmatrix} e_1 & \dots & e_m & & \\ m_{1,1} & \dots & m_{1,m} & \to & a_1 \\ \dots & \dots & \dots & \to & \dots \\ m_{n,1} & \dots, m_{n,m} & \to & a_n & \end{pmatrix}$$

Étant donné une expression $e = C\,(e_1, \dots, e_k)$, on définit des opérateurs d'accès au constructeur $\text{constr}(e) = C$ et au i-ème champ : $\#i(e) = e_i$. Un environnement est une fonction partielle des variables aux valeurs.

1. Définir une relation $m \leq_\sigma v$ établissant la compatibilité entre un motif m et une valeur v, étant donné un environnement σ.
2. On note $\text{Match}\,(v_1, \dots, v_m), M$ le résultat du filtrage de la matrice M sur $(v_1, \dots, v_m)$. Soit a_i une action, σ un environnement. Sous quelle(s) condition(s) a-t-on $\text{Match}\,((v_1, \dots, v_m), M) = (a_i, \sigma)$.
3. Décrire informellement comment éliminer le motif joker, sans ajouter de cas, sur l'exemple :

```
let rec len l=
        match l with
        | Empty -> 0
        | Cons(w, tl) -> 1 + (len tl)
```

   Définir cette transformation sur les matrices de filtrage. Donner un critère justifiant de la correction de la transformation.
4. Soit F une matrice de filtrage. On note $S(c, F)$ l'opérateur qui transforme la matrice de filtrage en supposant que e_1 commence par un constructeur c d'arité a. Illustrer le résultat de $S\,(\texttt{Cons}, F)$ sur la matrice du code ci-dessous :

```
let rec merge l1 l2 = match l1, l2 with
| Empty, l | l, Empty -> l
| Cons(h1, t1), Cons(h2, t2) ->
        if h1 < h2 then Cons(h1, merge t1 l2)
        else Cons(h2, merge l1 t2)
```

   Décrire la transformation opérée par $S(c, F)$. Donner un critère justifiant de la correction de la transformation.

Solution: Filtrage par Motif en OCaml

1. On donne :
   - $x \leq_\sigma v \Leftrightarrow \sigma(x) = v$
   - $\_ \leq_\sigma v$
   - $C(m_1, \dots, m_n) \leq_\sigma C'(v_1, \dots, v_o) \Leftrightarrow C = C' \wedge n = o \wedge \forall i, m_i \leq_\sigma v_i$
   - $(m_1 \mid m_2) \leq_\sigma v \Leftrightarrow m_1 \leq_\sigma v \vee m_2 \leq_\sigma v$.
2. Il faut que $\forall 1 \leq j < i, (v_1, \dots, v_m) \not\leq_\sigma (m_{j,1}, \dots, m_{j,m})$ et que $(v_1, \dots, v_m) \leq_\sigma (m_{i,1}, \dots, m_{i,m})$. Match est aussi défini si le match n'est jamais résolu, auquel cas on lève une exception.
3. On ajouter une variable qui n'est pas variable libre de l'action à droite de `->`. On note $NJ(M)$ cette transformation. `variables` est définie récursivement pour extraire les variables définies dans un motif :

$$NJ(M)_{i,j} = \begin{cases} M_{i,j} & \text{si } M_{i,j} \neq \_ \\ v & \text{tel que } v \notin \texttt{variables\_libres}(a_i) \wedge v \notin \{\texttt{variables}(m_{i,k} \mid k \neq j)\} \end{cases}$$

   Il suffit alors de prouver que $\text{Match}((v_1, \dots, v_m), M) = \text{Match}((v_1, \dots, v_m), NJ(M))$.
4. On a :

$$F = \begin{pmatrix} l1 & l2 & & \\ Empty & l & \to & l \\ l & Empty & \to & l \\ Cons(h1, t1) & Cons(h2, t2) & \to & \dots \end{pmatrix} \qquad S(Cons, F) = \begin{pmatrix} \#1(l1) & \#2(l1) & l2 & & \\ \_ & \_ & Empty & \to & l1 \\ h1 & t1 & Cons(h2, t2) & \to & \dots \end{pmatrix}$$

   Formaliser la transformation sur les indices n'est pas pratique. Voici les étapes :
   - On commence par changer $(e_1, \dots, e_m)$ en $(\#1(e_1), \dots, \#a(e_1), e_2, \dots, e_m)$
   - On raisonne par cas sur $m_{i,1}$ pour donner $M' \to A'$. Chaque ligne peut donner au plus deux lignes correspondantes dans la matrice de filtrage de $S(c, F)$:
     - Si $m_{i,1} = c(q_1, \dots, q_a)$ la ligne est conservée et étendue à gauche.
     - Si $m_{i,1} = c'(\_)$, la ligne est supprimée.
     - Si $m_{i,1} = v$ une variable, il n'y a rien à matcher dans l'extension à gauche et il faut étendre l'action en ajoutant un let pour lier la variable.
     - De même si $m_{i,1} = \_$.
     - Si $m_{i,1} = q_1, q_2$ on génère deux lignes, une pour q_1 et une pour q_2

   La correction est la même que précédemment.

Page 36 : `https://diplome.di.ens.fr/informatique-ens/annales/2024_Info-rapport.pdf`

Complexité, mais dur
Matthieu Boyer
Planche B

Khôlle MP2I
GRRR

Acupuncture

On dispose d'une paire stéréo de caméras O_L et O_R. On cherche à trouver la position réelle d'un point P étant données les coordonnées X_L et X_R de sa projection sur l'image de référence des deux caméras. Notant R, T la rotation et la translation entre les images de références, on a $X_R = R(X_L - T)$. On notera $S = T\wedge$ la matrice associée à l'application linéaire $X \mapsto T \wedge X$.

1. Montrez que $E = RS$ vérifie $X_R^T E X_L = 0$. On pourra utiliser sans démonstration que $R^T R = RR^T = I_3$.

Il nous suffit donc d'évaluer E pour retrouver les coordonnées de P. Remarquons que l'équation précédente tient pour tout X_R, X_L qui sont coplanaires avec O_L et O_R.

2. En écrivant y, y' comme des vecteurs de troisième coordonnée 1, et en introduisant les coordonnées de e, montrez que $y'^T E y = 0$ peut se réécrire comme $e \cdot \tilde{y} = 0$ où $\tilde{y} \simeq y' y^T$.
3. Étant donné un ensemble de points P_k qui correspondent à un ensemble de vecteurs $\tilde{y_k}$, réécrivez la contrainte ci-dessus comme un système linéaire homogène, c'est à dire $e^T Y = 0$.
4. Donnez un algorithme calculant e à partir de Y. On supposera qu'on dispose d'au moins 8 vecteurs linéairement indépendants $\tilde{y}_k$.

Solution: Acupuncture

Voir `https://en.wikipedia.org/wiki/Eight-point_algorithm`

1. On a :

$$\begin{aligned}(X_L - T)^T \wedge X_L &= 0\\ (X_L - T)^T R^T RT^l and X_L &= 0\\ X_R^T RT \wedge X_L &= 0\\ X_R^T E X_L &= 0\end{aligned}$$

2. La contrainte peut se réécrire comme

$$y'_1 y_1 e_{11} + y'_1 y_2 e_{12} + y'_1 e_{13} + y'_2 y_1 e_{21} + y'_2 y_2 e_{22} + y'_2 e_{23} + y_1 e_{31} + y_2 e_{32} + e_{33} = 0$$

   c'est à dire $e \cdot \tilde{y}$ où $\tilde{y}$ est la version flatten des 9 coefficients de $y' y^T$.
3. Si on dispose de n points qu'on pose comme colonnes d'une matrice Y, on a $e^T Y = 0$, la j-ème ligne du produit étant le produit scalaire $e \cdot \tilde{y}_j$.
4. On utilise le pivot de Gauss. Ce problème est celui-de trouver un vecteur propre associé à 0 pour Y, ce qui n'est uniquement déterminé que si on a au moins 8 points.

Complexité, mais dur
Matthieu Boyer
Planche C

Khôlle MP2I
GRRR

Complexité, mais dur
Matthieu Boyer
Planche C : Corrigés

Khôlle MP2I
GRRR