

Projet de recherche

Résumé de l'article **Handling Algebraic effects**

Jordan Ischard

Université d'Orléans

2020-2021

Sommaire

Introduction

Problème de l'impératif dans un langage fonctionnel
Proposition pour gérer les effets

Gestion des effets dans l'article

Validé des gestionnaires

Confrontation

Conclusion

Sommaire

Introduction

Problème de l'impératif dans un langage fonctionnel

Proposition pour gérer les effets

Gestion des effets dans l'article

Validé des gestionnaires

Confrontation

Conclusion

Ajout de principes impératifs dans un langage fonctionnel

But

Ajouter des fonctionnalités en plus dans les langages fonctionnels

Ajout de principes impératifs dans un langage fonctionnel

But

Ajouter des fonctionnalités en plus dans les langages fonctionnels

Problème

Ajoute des effets de bords que les langages fonctionnels purs non pas.

Ajout de principes impératifs dans un langage fonctionnel

But

Ajouter des fonctionnalités en plus dans les langages fonctionnels

Problème

Ajoute des effets de bords que les langages fonctionnels purs non pas.

Exemple

On peut avoir des effets de bords pour les appels mémoires ou encore les entrées/sorties.

Ajout de principes impératifs dans un langage fonctionnel

But

Ajouter des fonctionnalités en plus dans les langages fonctionnels

Problème

Ajoute des effets de bords que les langages fonctionnels purs non pas.

Exemple

On peut avoir des effets de bords pour les appels mémoires ou encore les entrées/sorties.

Idée

Créer une structure qui va gérer ces effets.

Sommaire

Introduction

Problème de l'impératif dans un langage fonctionnel

Proposition pour gérer les effets

Gestion des effets dans l'article

Validé des gestionnaires

Confrontation

Conclusion

Réponses déjà proposées

Les Monades

Eugenio Moggi a proposé le principe de **Monade** pour gérer les effets.

Réponses déjà proposées

Les Monades

Eugenio Moggi a proposé le principe de **Monade** pour gérer les effets.

Les Monades en Haskell

```
1 instance Monad Maybe where
2   return x = Just x
3
4   Nothing >>= f = Nothing
5   Just x >>= f  = f x
```

Réponses déjà proposées

Les Monades

Eugenio Moggi a proposé le principe de **Monade** pour gérer les effets.

Les Monades en Haskell

```
1 instance Monad Maybe where
2   return x = Just x
3
4   Nothing >>= f = Nothing
5   Just x >>= f = f x
```

Les théories d'équations

Plotkin et Power ont proposé des **opérations** comme source des effets et une **théorie d'équation** pour décrire leurs propriétés.

Réponses déjà proposées

Les Monades

Eugenio Moggi a proposé le principe de **Monade** pour gérer les effets.

Les Monades en Haskell

```
1 instance Monad Maybe where
2   return x = Just x
3
4   Nothing >>= f = Nothing
5   Just x >>= f = f x
```

Les théories d'équations

Plotkin et Power ont proposé des **opérations** comme source des effets et une **théorie d'équation** pour décrire leurs propriétés.

Un effet Algébrique

C'est un effet qui peut être représenté par les deux modèles proposés.

Sommaire

Introduction

Gestion des effets dans l'article

Intuition et principe

Mise en place dans une syntaxe

Exemple

Validé des questionnaires

Confrontation

Conclusion

Sommaire

Introduction

Gestion des effets dans l'article

Intuition et principe

Mise en place dans une syntaxe

Exemple

Validé des questionnaires

Confrontation

Conclusion

Comment gérer les effets ?

Intuition

Reprendre le principe de Plotkin et Power et l'associer aux travaux de Benton et Kennedy sur les gestionnaires.

Comment gérer les effets ?

Intuition

Reprendre le principe de Plotkin et Power et l'associer aux travaux de Benton et Kennedy sur les gestionnaires.

Principe

Les effets vont avoir pour source des **opérations** et vont être géré par un **tableaux associatifs (map)** contenu dans un gestionnaire.

De plus, on ajoute une **théorie d'effet** qui explicite les propriétés des effets.

Sommaire

Introduction

Gestion des effets dans l'article

Intuition et principe

Mise en place dans une syntaxe

Exemple

Validé des questionnaires

Confrontation

Conclusion

Qu'ajoute-on pour mettre en place la gestion des effets ?

Syntaxe ajouté

On part de la syntaxe proposé par Levy dans son article sur *call-by-push-value*.

source de l'effet : $\mathbf{op}_V(x.M')$

structure de gestion : M **handled with** H *to* $x.N$

gestionnaire : $\{\mathbf{op}_z(k) \mapsto M_{\mathbf{op}}\}_{\mathbf{op}}$

Qu'ajoute-on pour mettre en place la gestion des effets ?

Syntaxe ajouté

On part de la syntaxe proposé par Levy dans son article sur *call-by-push-value*.

source de l'effet : $\mathbf{op}_V(x.M')$

structure de gestion : M **handled with** H to $x.N$

gestionnaire : $\{\mathbf{op}_z(k) \mapsto M_{\mathbf{op}}\}_{\mathbf{op}}$

Fonctionnement

On reprend la strucutre de gestion ci-dessus avec $\mathbf{op}_V(y.M') \in M$ et $\{\mathbf{op}_z(k) \mapsto M_{\mathbf{op}}\} \in H$.

$M_{\mathbf{op}}[V/z, M'[W/y]$ **handled with** H to $x.N/k(W)$

Si l'opération n'est pas géré par H alors elle se **gère elle-même**.

Sommaire

Introduction

Gestion des effets dans l'article

Intuition et principe

Mise en place dans une syntaxe

Exemple

Validé des questionnaires

Confrontation

Conclusion

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\mathbf{get}_z(k) \mapsto k(n)\}$$

Prenons l'expression suivante :

let n be 20 in

$\mathbf{get}_l(x.\mathbf{get}_l(y.\mathbf{return} \ x + y))$

handled with $H_{temporary}$ to $z.\mathbf{return} \ z + 2$

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\mathbf{get}_z(k) \mapsto k(\textcolor{red}{n})\}$$

Prenons l'expression suivante :

let n be 20 in

get_l(x.get_l(y.return $x + y$))

handled with $H_{temporary}$ to z.return $z + 2$

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\text{get}_z(k) \mapsto k(20)\}$$

Prenons l'expression suivante :

get_l(**x.get_l**(**y.return** $x + y$))

handled with $H_{temporary}$ to **z.return** $z + 2$

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\text{get}_z(k) \mapsto k(20)\}$$

Prenons l'expression suivante :

get_l(y.**return** 20 + y)

handled with *H_{temporary}* **to** z.**return** z + 2

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\mathbf{get}_z(k) \mapsto k(20)\}$$

Prenons l'expression suivante :

return 20 + 20

handled with $H_{temporary}$ **to** **z.return** z + 2

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\mathbf{get}_z(k) \mapsto k(20)\}$$

Prenons l'expression suivante :

return 40 + 2

dérogation à la lecture seule

Valeur Temporaire sans modification dans la mémoire

Le gestionnaire définit pour l'effet est le suivant :

$$H_{temporary} = \{\mathbf{get}_z(k) \mapsto k(20)\}$$

Prenons l'expression suivante :

return 40 + 2

La réponse

C'est 42 !

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Quand est-ce qu'un questionnaire est valide ?

Questionnaire générique indéfinissable

Confrontation

Conclusion

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Quand est-ce qu'un questionnaire est valide ?

Gestionnaire générique indécidable

Confrontation

Conclusion

Quand est-ce qu'un gestionnaire est valide ?

Comment exprimer la validité ?

La validité d'un gestionnaire est exprimé à travers l'assertion d'existence, d'égalité et d'inégalité de *Kleene*.

Quand est-ce qu'un gestionnaire est valide ?

Comment exprimer la validité ?

La validité d'un gestionnaire est exprimé à travers l'assertion d'existence, d'égalité et d'inégalité de *Kleene*.

La validité d'un gestionnaire

On l'exprime de cette manière :

$$H \downarrow \Leftrightarrow \bigwedge \{ \forall x. M_{\text{op} \downarrow} \} \wedge \bigwedge \{ T_1^H \simeq T_2^H \mid \Gamma \mid Z \vdash T_1 = T_2 \in \tau \}$$

Quand est-ce qu'un gestionnaire est valide ?

Comment exprimer la validité ?

La validité d'un gestionnaire est exprimé à travers l'assertion d'existence, d'égalité et d'inégalité de *Kleene*.

La validité d'un gestionnaire

On l'exprime de cette manière :

$$H \downarrow \Leftrightarrow \bigwedge \{ \forall x. M_{\text{op}} \downarrow \} \wedge \bigwedge \{ T_1^H \simeq T_2^H \mid \Gamma \mid Z \vdash T_1 = T_2 \in \tau \}$$

La validité d'une structure de gestion

$$M \text{ handled with } H \text{ to } x.N \downarrow \Leftrightarrow M \downarrow \wedge H \downarrow \wedge \forall x. N \downarrow$$

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Quand est-ce qu'un questionnaire est valide ?

Questionnaire générique indéfinissable

Confrontation

Conclusion

Les gestionnaires en général sont indécidable

Limite des gestionnaires

En général la validité des gestionnaires est **indécidable**.

Les gestionnaires en général sont indécidable

Limite des gestionnaires

En général la validité des gestionnaires est **indécidable**.

Comment palier à se problème

Deux approches sont possibles :

1. Obliger l'utilisateur a utiliser des gestionnaires que l'on sait valide (donc prédéfinis dans le langage) : **la responsabilité est au créateur du langage**
2. Laisser l'utilisateur faire comme il le souhaite tout en lui laissant la possibiliter de faire des gestionnaires invalide : **la responsabilité est à l'utilisateur.**

Les gestionnaires en général sont indécidable

Limite des gestionnaires

En général la validité des gestionnaires est **indécidable**.

Comment palier à se problème

Deux approches sont possibles :

1. Obliger l'utilisateur a utiliser des gestionnaires que l'on sait valide (donc prédéfinis dans le langage) : **la responsabilité est au créateur du langage**
2. Laisser l'utilisateur faire comme il le souhaite tout en lui laissant la possibiliter de faire des gestionnaires invalide : **la responsabilité est à l'utilisateur.**

Vers des gestionnaires plus simple

Il est possible de trouver un sous-ensemble de gestionnaire qui est **décidable**.

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Confrontation

Les divergences

Exemple de conversion entre langages

Difficulté d'implémentation

Conclusion

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Confrontation

Les divergences

Exemple de conversion entre langages

Difficulté d'implémentation

Conclusion

Motivation et fonctionnement divergent

Langage de l'article	Langage erpl
<ul style="list-style-type: none">○ Créer pour voir l'aspect mathématique de la gestion d'effets algébriques	<ul style="list-style-type: none">○ Créer pour avoir une version allégé du langage Ocaml

Motivation et fonctionnement divergent

Langage de l'article	Langage erpl
<ul style="list-style-type: none">○ Créer pour voir l'aspect mathématique de la gestion d'effets algébriques○ La source d'un effet est une opération	<ul style="list-style-type: none">○ Créer pour avoir une version allégé du langage Ocaml○ La source d'un effet est un type

Motivation et fonctionnement divergent

Langage de l'article	Langage erpl
<ul style="list-style-type: none">○ Créer pour voir l'aspect mathématique de la gestion d'effets algébriques○ La source d'un effet est une opération○ Activation implicite de l'effet	<ul style="list-style-type: none">○ Créer pour avoir une version allégé du langage Ocaml○ La source d'un effet est un type○ Activation explicite de l'effet

Motivation et fonctionnement divergent

Langage de l'article	Langage erpl
<ul style="list-style-type: none">○ Créer pour voir l'aspect mathématique de la gestion d'effets algébriques○ La source d'un effet est une opération○ Activation implicite de l'effet○ Gestionnaire global implicite car l'activation est implicite	<ul style="list-style-type: none">○ Créer pour avoir une version allégé du langage Ocaml○ La source d'un effet est un type○ Activation explicite de l'effet○ Pas de gestionnaire global, tout activation sans gestionnaire provoque une erreur

Table: Comparaison entre les deux langages

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Confrontation

Les divergences

Exemple de conversion entre langages

Difficulté d'implémentation

Conclusion

Passage du langage de l'article vers **erpl**

Reprise de l'exemple précédent

let n **be** 20 **in**

get_I(x .**get**_I(y .**return** $x + y$))

handled with $\{\text{get}_I(k) \mapsto k(n)\}$ **to** z .**return** $z + 2$

Passage du langage de l'article vers **erpl**

Reprise de l'exemple précédent

let n **be** 20 **in**

get_I(x .**get**_I(y .**return** $x + y$))

handled with $\{\text{get}_I(k) \mapsto k(n)\}$ **to** z .**return** $z + 2$

Reprise de l'exemple converti en **erpl**

```
1 type memory = Get of 'a ref -> 'a
2               | Set of 'a ref * 'a -> unit;;
3
4 let get = Get(fun x -> !x) and n = 20 in
5     handle (fun x -> fun y -> x + y + 2)
6         ((perform get) l) ((perform get) l)
7     with (Get f), k -> k (fun x -> n) ;;
8
```

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Confrontation

Les divergences

Exemple de conversion entre langages

Difficulté d'implémentation

Conclusion

Une histoire d'appels systèmes

Comment ça fonctionne ?

En prenant, le principe d'opération comme source d'effet on implique une gestion des opérations particuliers.

Quand on a pas de gestionnaire, les opérations se *gèrent elles-même*.

Une histoire d'appels systèmes

Comment ça fonctionne ?

En prenant, le principe d'opération comme source d'effet on implique une gestion des opérations particuliers.

Quand on a pas de gestionnaire, les opérations se *gèrent elles-même*.

Idée

Créer un gestionnaire globale implicite ajouté à la compilation.

Une histoire d'appels systèmes

Comment ça fonctionne ?

En prenant, le principe d'opération comme source d'effet on implique une gestion des opérations particuliers.

Quand on a pas de gestionnaire, les opérations se *gèrent elles-même*.

Idée

Créer un gestionnaire globale implicite ajouté à la compilation.

Problèmes d'appels systèmes

Cela veut dire que notre gestionnaire globale doit être capable de faire des appels systèmes et donc que le langage en soit capable.

Pas facilement intégrable !

Sommaire

Introduction

Gestion des effets dans l'article

Validé des questionnaires

Confrontation

Conclusion

Que peut-on conclure ?

TODO : No se