





# Vérification statique des séquences d'appels aux fonctions collectives MPI

Clément GAVOILLE - Jean Baptiste SKUTNIK

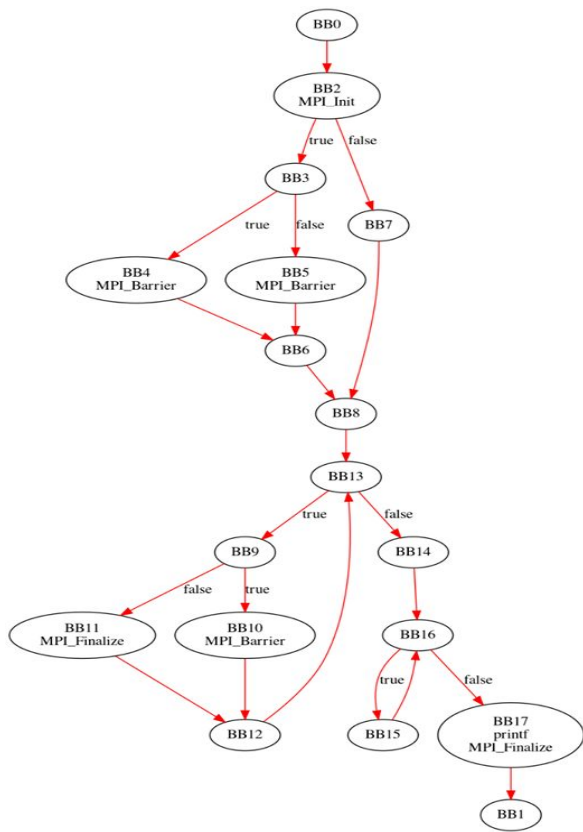
Projet de Compilation Avancée  
ENSIIE 2019

- 
- Objectifs du projet
  - Détection des appels aux fonctions collectives
  - Détermination des divergences
  - Détermination des noeuds à risque
  - Gestion des directives
  - Conclusion

- 
- Mise en place d'une passe émettant un warning si le programme compilé peut être bloqué par un deadlock dû à un appel d'une fonction collective MPI;
  - Définition d'une directive permettant de sélectionner les fonctions analysées par la passe.


On utilise l'API de GCC pour répondre à ces problèmes.

# Détection des appels aux fonctions collectives



- On travaille sur les *basic blocks* du Control Flow Graph (CFG).
- On itère ensuite sur les statement *gimple* de chaque blocs.
- On sépare les blocs contenant plusieurs appels.
- Notre liste de fonctions cibles est définie dans **MPI\_collectives.def**

# Détermination des divergences

- 
- Utilisation de la *post-dominance frontier* (PDF) pour repérer les noeuds à risque.

$$PDF(N) = \{z \mid z \in \cup_{n \in N} PDF(n), z \in \cup_{k \in \bar{N}} PDF(k)\}$$

- Travail sur les ensembles de noeuds ayant un appel à la même fonction collective MPI.
- Calcul de la PDF de chaque ensemble d'appels aux fonctions collectives MPI.

PDF non vide => Ensemble de noeuds possiblement à risque

## Détermination des noeuds à risque



- On effectue un parcours en profondeur du CFG à partir des noeuds de la PDF tout en ignorant les boucles.
- On obtient ainsi une séquence des appels aux fonctions MPI de chaque chemin partant de ce noeud vers la destination.
- Si on détecte deux séquences différentes, il existe donc deux chemins différents n'effectuant pas d'appels aux fonctions cibles.
- Il y a donc possibilité de deadlock à l'exécution.
- On affiche donc un *warning* renvoyant la dernière ligne du ou des noeuds à risque.

- 
- Format choisi des directives :

*#pragma mpicoll check f1*

*#pragma mpicoll check(f1,f2)*

- Elles permettent à l'utilisateur de choisir les fonctions analysées lors de la passe.
- Elles ne peuvent pas être définies dans un corps de fonction, spécifier plusieurs fois la même fonction ou préciser une fonction n'étant pas présente dans le code source.



## Travail effectué :

- Mise en place d'une passe permettant de détecter des cas de deadlock possibles dû à une mauvaise gestion des appels aux fonctions collectives MPI.
- L'utilisateur peut spécifier les fonctions à traiter à l'aide de *#pragma*.

## Ouverture possible :

- Meilleure gestion des boucles;
- Possibilité d'effectuer une analyse inter-procédurale;
- Stopper le programme à l'exécution en cas de deadlock.





# Merci pour votre attention

Clément GAVOILLE - Jean Baptiste SKUTNIK

Projet de Compilation Avancée  
ENSIIE 2019