# Compilation optimisante vérifiée sur forme SSA

(stage de L3)

Delphine Demange
delphine.demange@irisa.fr
Université de Rennes 1 / IRISA / Inria Rennes

David Pichardie
david.pichardie@irisa.fr
ENS Rennes / IRISA / Inria Rennes

**Présentation générale du domaine** Un compilateur vérifié est un compilateur pour lequel il a été démontré formellement que, pour tout code source, le code cible généré et le code source se comportent de la même façon. Une telle démonstration est rendue possible grâce à l'emploi d'un assistant à la preuve, un environnement de programmation dans lequel il est possible, grâce au système de types très riche du language, de définir des programmes et de prouver formellement des propriétés de correction à son égard. Développer une telle preuve pour un compilateur réaliste est un travail considérable, mais le travail de Xavier Leroy [1] a déjà montré sa faisabilité pour un compilateur réaliste du langage C, CompCert, en utilisant l'assistant à la preuve Coq.

**Objectifs du stage** Ce résultat scientifique donne beaucoup d'espoirs sur ce que l'on peut réussir à démontrer formellement dans une chaîne de compilation. Cependant, CompCert ne s'appuie pas sur une technique de compilation majeure : la forme SSA (Single Static Assignment form). Un programme en forme SSA a pour propriété qu'une même variable est affectée une et une seule fois dans le texte du programme. De nombreuses optimisations spécifiques à cette représentation ont été développées.

Le but de ce stage est d'étendre le compilateur CompCertSSA [2], développé dans l'équipe Celtique à Rennes, qui ajoute une forme SSA à CompCert. Actuellement, une seule optimisation SSA est réalisée dans le compilateur et l'objectif de ce stage sera d'ajouter une des optimisations classiques en forme SSA, telles que Sparse Conditional Constant Propagation [3] ou Partial Redundancy Elimination [4]. Il conviendra donc de programmer et de prouver correct dans l'assistant de preuve Coq l'une de ces optimisations et de mesurer son impact sur la qualité du code généré par le compilateur CompCertSSA.

**Compétences espérées** Le candidat devra exprimer de l'intérêt pour la compilation, la preuve rigoureuse, la programmation fonctionnelle et le travail en équipe. Aucune expérience préalable avec l'assistant de preuve Coq n'est requise.

**Références bibliographiques**

[1] X. Leroy. Formal verification of a realistic compiler. X. Communications of the ACM, 52(7) :107-115, 2009.

[2] G. Barthe, D. Demange, and D. Pichardie. A formally verified SSA-based middle-end - Static Single Assignment meets CompCert. In Proc. of 21th European Symposium on Programming (ESOP), LNCS. 2012.

[3] M. N. Wegman and F. K. Zadeck. Constant Propagation with Conditional Branches. ACM Transactions on Programming Languages and Systems, 13(2), April 1991, pages 181-210.

[4] R. Kennedy, S. Chan, S.M. Liu, R. Lo, T. Peng and F. Chow. Partial Redundancy Elimination in SSA Form. ACM Transactions on Programming Languages Vol. 21, Num. 3, pp. 627-676, 1999.