**Research Statement** .

My research interests are in the broad area of program analysis and code transformation. Currently I am working on the problem of eliminating redundancies based on value equivalence of expressions. There are two main aspects to this problem: detection of value equivalent expressions and elimination of redundancies. Though work has been done in these two areas, I strongly feel, there is scope for further research. The existing algorithms for value based redundancy elimination, I found, are not "simple" to comprehend and some of them miss certain redundancies. Through my research, I hope and strive for, new "simple" and "aggressive" method for detection of value equivalent expressions and elimination of redundancies.

Currently my focus is on the first step, that is, detecting value equivalent expressions in programs. Two prominent algorithms for detecting value equivalent expressions are the ones proposed by Kildall [2] and Alpern et al [1]. Kildall's algorithm detects equivalences in non-SSA form of programs and is Herbrand complete. Alpern et al's algorithm works on SSA form of programs and is efficient compared to Kildall's. However it misses out on detecting certain (Herbrand) equivalences. Based on my literature survey, I have observed that most of the works in detecting equivalences assume SSA form of programs. This I feel is motivated by efficiency aspect. However, most of these algorithms are not Herbrand complete. One possible line of thought I have is to design a Herbrand complete algorithm for detecting equivalences, considering efficiency aspect, in SSA form of programs.

Another possible line of thought is to have a completely different perspective to the problem of detecting value equivalent expressions. The existing works in the literature view this problem as simply detecting value equivalent expressions in programs. However, I feel, we can view the problem as detecting lexically equivalent expressions. This is based on the idea that it may be possible to map the problem of detecting value equivalent expressions to the problem of detecting lexically equivalent expressions.

Once the equivalences are detected, the next step is to detect and eliminate redundancies based on value equivalence. During my Master's thesis work, I was working on a different version of this problem - elimination of partially redundant expressions *based on lexical equivalence*. This classical program transformation, known as Partial Redundancy Elimination (PRE), is weaker compared with value based redundancy elimination since it eliminates redundancies based on lexical equivalence of expressions rather than the value equivalence. One possible line of thought I have for redundancy elimination is to look for (value based) redundancies *along a program path* and then eliminate them.

The current work in value based redundancy elimination, I feel, can even lead to a unified algorithm to perform redundancy elimination, constant propagation, and copy propagation. My research in the area of program analysis, even though currently for the purpose of program transformation, has applications in many other areas such as program security, translation validation, plagiarism detection, and program verification.

It is my desire to make some fundamental contributions in the area of program analysis and possible applications.

# References

[1] Alpern, B., Wegman, M. N., and Zadeck, F. K. Detecting Equality of Variables in Programs. In *Proceedings of the Symposium on Principles of Programming Languages*, pp 1-11, January 1988.

[2] Kildall, Gary A. A unified approach to global program optimization. In *Proceedings of the Symposium on Principles of Programming Languages*, pp 194-206, 1973.