

Connection Analysis : A Practical Interprocedural Heap Analysis for C

- Rakesh Ghiya and Laurie Henderson.

Presented by,
Niranjan K V Viswanath.



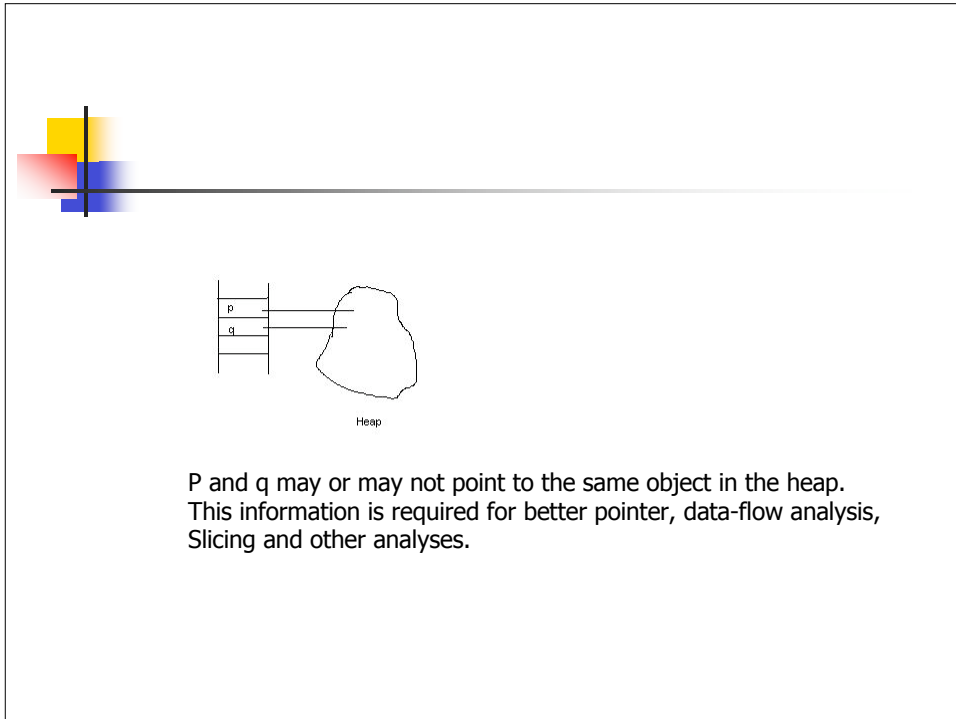
Problems addressed by the Paper.

- **Big picture problem.**
Program analysis and optimization are required since we always want efficiency and to minimize the runtime.
- **Specific problem.**
Pointer analysis are not precise enough if it does not have proper heap analysis. Heap analysis is an undeterministic problem.
- **Problem Setup.**

Consider the following example:

```
*p=  
=*q
```

What does the heap look for these statements



Context of the problem

'C' language is used to a large extent which make use of a lot of pointers, thus their analysis is important for better program analysis.

Erroneous pointer usage causes a lot of errors in programs.
Finding out errors due to improper pointer usage is a difficult problem.

A lot of work has been done in heap modeling techniques like [JM81, JM82, LH88 etc] and heap modeling implementation methods [LR92, CBC93, EGH, WL95].

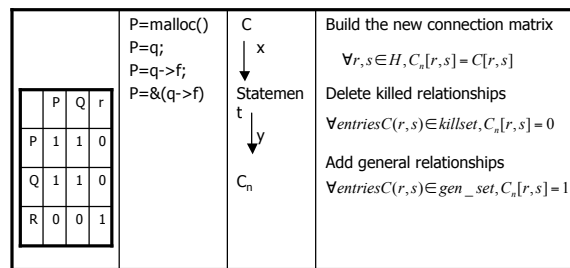
Just for the records a separate thesis [GHI95] has a discussion and comparison of all implementation methods.

Approach

A connection matrix is created which shows the connectivity of heap objects.

A new connection matrix is computed with the help of values from 'kill_set' and 'Gen_set'

The first box represents a connection matrix. The second box represents sample code. Third shows conversion to a new connection matrix. Fourth box shows the operations on the kill and gen sets.



Conclusions

- A complete context-sensitive analysis is done with reasonable cost since the invocation graphs do not explode.
- Approach leads to a faster more efficient points-to analysis.
- Connection analysis can be used to distinguish between different linked data-structures.
- If programs use dis-joint data structures , connection matrix analysis provides more accurate information for resolving heap directed indirect references.



Future research questions

- Develop better memory management techniques for handling large programs with large invocation graphs.
- Build automatic tools using connection matrix analysis to provide good analyses.
- Research to be done on how to get a better trade-off between efficiency vs time/space requirements for pointer analysis and program analysis in general.



Critique

- Could have justified the effectiveness of their Analysis better.
- Empirical comparisons have not been provided versus other approaches to show the effectiveness of their approach.



Relation to CS 653

- Shows the various reasons which hinder efficient pointer and heap analysis.
- Describes a new approach to tackling the heap analysis.
- Personally, I feel that the paper will make a strong impact on pointer analysis implementation.