The impact of five orthogonal analysis parameters has been studied:

- **Context-sensitivity:** with vs. without
- **Retaining temporary variables for additional names vs. removing them**
- **Alias test:** lazy vs. extensive
- **Alias test:** common tail vs. extensive
- **With context-sensitivity the many invocations of a function are kept separate during analysis; functions are analysed once for every calling context. Clearly, this is more precise than a context-insensitive analysis, but also more costly in general.**

### COMPARISON

#### SRW

The shape analysis by Sagiv, Smolensky, and Wilhelms was the first shape analysis to achieve strong updates for languages with destructive updating. It uses static shape graphs to approximate the structure of the heap.

```c
int main(int argc, char **argv) {
    List *p = a->next;
    t = NULL;
    a->next = t;
    int x = p->next->value;     // Stmt C
    return 0;
}
```

#### NNH

The shape analysis described by Nielsen, Nielsen, and Hankin in “Principles of Program Analysis” is based on the SRW analysis but uses sets of compatible shape graphs instead of a single graph. This makes the analysis more precise but also computationally more expensive.

```c
int main(int argc, char **argv) {
    List *a = new List(2);
    t->next = new List(3);
    return 0;
}
```

#### RWS02

Interpreting shape graphs naively (as described in RWS02):  - expressions leading to the same named node are must-aliases  - expressions leading to the summary location are may-aliases

```c
int main(int argc, char **argv) {
    List *a = new List(2);
    t = new List(3);
    return 0;
}
```

#### ANALYSIS RESULTS

- When one representation is connected to the other graphs are often equal through the graphs the analyses may not be equal

#### Context

A context-insensitive analysis merges the information available at different call sites of a function. It analyses the body of each function only once for all calling contexts combined and returns the merged information to all call sites. With context-sensitivity the many invocations of a function are kept separate during analysis; functions are analysed once for every calling context. Clearly, this is more precise than a context-insensitive analysis, but also more costly in general.

### SUMMARY

The effects of five orthogonal analysis parameters have been studied:

- **Shape analysis algorithm:** SRW vs. NNH
- **Context-sensitivity:** with vs. without
- **Retaining temporary variables for additional names vs. removing them**
- **Alias test:** lazy vs. extensive
- **Alias test:** common tail vs. extensive

#### Results

- Ignoring context information produced the worst results
  - naively: fast but aliased data is bad; always use context information
- SRW shape analysis: extensive test cannot be more precise than the lazy test on SRW graphs — always perform the lazy test to save time!
  - precise NNH: fast but aliased data is bad; always use context information
  - without tempor or common tail extension, SRW and NNH have comparable precision, but SRW is roughly 5x faster
  - retaining temporary variables increased precision for both SRW and NNH (23% and 21% smaller aliased sets) but also increased graph class and therefore analysis runtime (5x — expensive precision)
  - common tails increased precision only in combination with NNH, but then at no measurable increased cost

#### Costs

- fastest: SRW + context + lazy — common tails — temporaries
- most precise: NNH + context + extensive + common tails

#### COMPARISON A

- Results of different shape analyses are hard to compare directly
- When one representation is connected to the other graphs are often equal through the graphs the analyses may not be equal

#### COMPARISON B

- Interpreting Shape Graphs to obtain alias sets makes different shape analyses comparable by the size of derived may-aliases sets
- Smaller may-alias sets are the better results and indicate a more precise underlying shape analysis

Kontakt: e0425543@student.tuwien.ac.at