Total Pasta: Unfailing Pointer Programs

Neil Mitchell, ndm AT cs.york.ac.uk
Department of Computer Science, University of York
Pasta – Linked List Example

```
list { 
  nil();
  cons(int head, ptr tail);
}

-- inserts an element into an ordered list
insert(int i, ptr s) {
  while (s::cons && s->head < i) s = s->tail;
  if (s::nil || s->head > i) *s = *cons(i, copy(s));
}

main() {
  ptr r = nil();
  insert(1,r); insert(9,r);
  insert(2,r); insert(8,r);
}
```
Total Pasta Functions?

- Must not crash
  - `if (s::nil) s = s->tail;`
- Must terminate
  - `while (s::cons) s = s;`
- Don't need to worry about
  - arithmetic overflow (no addition in Pasta!)
  - recursion (also not in Pasta)
- Assume unbounded memory
Subtype checking

- Subtype annotations
  - `if (x::cons) ...`
- Subtype assertions
  - `x->tail` requires `x::cons`
- Can use powerset to represent subtypes
  - `Subtype(x) ∈ \{\{cons,nil\}, \{nil\}, \{cons\}, ∅\}`

Type assertions can be discharged by static checking
Termination Checking

- Only has a `while` statement to loop
- There must be one variable that is advanced down an acyclic path during every iteration
  ```c
  while (s::cons) s = s->tail;
  ```
- Requires an acyclic annotation
  ```c
  list acyclic(tail) { ... }
  ```
My Approach

- B/Z inspired approach
  - Define postconditions for safety
  - Propagate backwards
  - Show the conditions are satisfied

- The Method
  - Assign a postcondition of True
  - Transform post conditions to generate preconditions
  - Total function has precondition of True
Details: Safe and Prec

- **Safe(\(\alpha\))** – the conditions for \(\alpha\) to be safe
  - \(\text{Safe}(\text{tail}) = s::\text{cons}\)

- **Prec(\(\alpha, \beta\))** – the condition \(\beta\), with \(\alpha\)
  - \(\text{Prec}(x = y, x::\text{cons}) = y::\text{cons}\)
  - \(\{y::\text{cons}\} x = y \{x::\text{cons}\}\)
Flow Structures (if)

- \( \{ \alpha \} \text{ if } (\text{cond}) \ t; \text{ else } f; \ {\beta} \)
- \( \alpha = \text{safe}(\text{cond}) \land \)
  \( (\text{cond} \implies \text{safe}(t) \land \text{prec}(t, \beta)) \land \)
  \( (\neg \text{cond} \implies \text{safe}(f) \land \text{prec}(f, \beta)) \)
A small example

```c
if (s : : nil || s->head > i )
    {True} *s = *cons(i, copy(s)); {True}
{True}
```

- Now lets expand the `||` ...

Expanding out the `||`

\[
\{(s::\text{nil} \Rightarrow \text{True}) \land (\neg s::\text{nil} \Rightarrow s::\text{cons})\}
\]

\[
i f \quad \{\text{True}\} \ (s::\text{nil} \ l)
\]
\[
\{\text{True}\} \ \text{stmt};
\]

\[
\ \text{else} \quad \{s::\text{cons}\} \ i f \quad \{s::\text{cons}\} (s->\text{head} > i)
\]
\[
\{\text{True}\} \ \text{stmt};
\]

Equivalent to: \{\text{True}\}
Ingredients of Checking

- Prec and Safe functions
- A predicate solver
- Fixed pointing for loops
- Check that acyclic property is preserved
- Check all loops terminate
Back to the example

- The precondition to main is True
- The precondition to insert is True
- Both are total functions

- Also tested on Queues, Binary Trees, 234 Trees, for insertion and deletion
  - Proves all to be total functions
Future Work

- Use a mainstream language, i.e. C++
- Extend Pasta with static typing, arithmetic
- Operate on individual procedures
  - Currently it expands them ALL inline
- Make it go faster
  - Some runs took hours (insert in 234 Tree)
  - Profiling gave 20x speedup with ease
Total Pasta: Unfailing Pointer Programs

Neil Mitchell, ndm AT cs.york.ac.uk
Department of Computer Science, University of York
Starred Assignment

\[ *a = *c \]

Notice that the value of \( b \) changes, without being mentioned.