NUI MAYNOOTH
Ollscoil na hÉireann Má Nuad

IRISH RESEARCH COUNCIL
An Chomhairle um Thaighde in Éirinn

Marie Farrell

REFINEMENT

WHAT IS REFINEMENT?

- Refinement provides a way for us to model software at different levels of abstraction
- We often start with a high level abstract specification and through a series of "Refinement steps" we develop a concrete implementation of the system at hand

MORGAN – "ON THE REFINEMENT CALCULUS"

- Notation and rules for deriving programs from their specifications
- Within a single formalism
- Based on weakest precondition: For a program P and predicate R over the program variables the "weakest precondition" is written
    - wp(P,R)
  - This is intended to describe exactly those states from which execution of P is guaranteed to establish R.

SPECIFICATIONS

- We specify a program P by giving both a pre-condition (pre) and a post-condition (post):
  - $pre \Rightarrow wp(P, post)$
- If ‘pre’ is true, then execution of ‘P’ must establish ‘post’

REFINEMENT

- We write:

$$[pre, post] \sqsubseteq P$$

to denote “the specification $[pre, post]$ is refined by P”

- **Definition:**

  For programs P and Q, we say that P is refined by Q written $\mathrm{P} \sqsubseteq Q$, iff for all post-conditions $post$:

$$wp(P, post) \Rightarrow wp(Q, post)$$

- Operationally $\mathrm{P} \sqsubseteq Q$ whenever Q resolves non-determinism in P, or terminates when P might not

- Therefore an NFA is refined by a DFA.

EXAMPLE

$$\begin{array}{l} if\ a \leq b \rightarrow a \coloneqq a - b \\ \square\ b \geq a \rightarrow b \coloneqq b - a \\ \hfill fi \end{array}$$

$$\sqsubseteq$$

$$\begin{array}{l} if\ a \leq b \rightarrow a \coloneqq a - b \\ \square\ a \nleq b \rightarrow b \coloneqq b - a \\ \hfill fi \end{array}$$

LAWS OF REFINEMENT

1. Weakening the precondition
   - More robust than previous
2. Strengthening the postcondition
   - Allows less choice than previous
3. Restricting change
   - Can change fewer variables than previous
4. Introducing fresh local variables
5. Introducing abort
6. Introducing skip
7. Introducing assignment
8. Introducing sequential composition
9. Introducing alternation
10. Introducing iteration

MORRIS – "STEPWISE REFINEMENT"

- Views programming as constructing a sequence of specifications, each one better defined than, but preserving the meaning of its predecessors; the final specification is a program in the language
- Note: the specifications arising in the construction of a program form a monotonic sequence

PRESCRIPTIONS

- A prescription $P||Q$ specifies a mechanism that when executed in a state satisfying P will terminate in a state satisfying Q
  - P and Q are predicates

REFINEMENT

- We proceed from the initial prescription $P||Q$ through a sequence of specifications s_i such that:

$$P||Q \sqsubseteq\ s_1 \sqsubseteq\ s_2 \sqsubseteq \ldots \sqsubseteq s_i$$

RULES OF REFINEMENT

- Given $P||Q$ there are 6 ways of choosing s such that

$$P||Q \sqsubseteq s$$

1. Skip
2. Assignment
3. Prescription
4. If statement
5. Composition
6. Block

EXAMPLES – IS THIS REFINEMENT

- Context Free Grammar
  - Morgan – yes because moving through a CFG reduces non-determinism
  - Morris – yes because it follows a less-defined, better-defined structure
- Compiler
  - Morgan – yes because no check is made against specifying a specification too much and therefore unproductive refinement steps may go unnoticed
  - Morris - not sure