Contract-based Software Development

Rasmus Guldborg Pedersen

January 2015

Overview

Invariants for loops and code contracts

What is an invariant for a loop, and how can it be used to reason about the behavior of a loop? Briefly explain what Code Contracts (.net tool) is and explain how it can be used to decorate a loop with contracts in order to ensure that a program assertion is an invariant.

Loop Invariant

```
// { Q }
// S0
// { P }
while(B) {
    // { P ∧ B }
    // S
    // { P }
}
// {P ∧ ¬ B ⇒ R }
```

Loop Invariant: Example

Algorithm for summing integers in a array.

$a[0] + a[1] + \ldots a[N-1] = (\Sigma i | 0 \leq i < N : a[i])$

Loop Invariant: Example

```
// { 0 ≤ N }
int n = 0;
int s = 0;
// { s = (Σ i | 0 ≤ i < n : a[i]) }
while (n != N) {
    // { s = (Σ i | 0 ≤ i < n : a[i]) ∧ n ≠ N }
    s = s + a[n];
    n = n + 1;
    // { s = (Σ i | 0 ≤ i < n : a[i]) }
}
// { s = (Σ i | 0 ≤ i < N : a[i]) ∧ n = N }
```

Loop Invariant: Example proof

Basis: $n = 1$

$a[0] = (\Sigma i | 0 \leq i < 1 : a[i])$

Inductive step: $n + 1$

$a[0] + a[1] + \ldots + a[n-1] + a[n] = (\Sigma i | 0 \leq i < n + 1 : a[i])$

Loop Invariant: Example proof

```
while (n != N) {
    s = s + a[n];
    // { s = (Σ i | 0 ≤ i < n + 1 : a[i]) }
    n = n + 1;
}
```

Loop Invariant: Termination

Function T such that loop execution ends when $T = 0$.
$T = N - n$ for the example.

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis
- Documentation

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis
- Documentation
- Runtime checking

Example using Code Contracts

Initialize and array a with value v.

Example: Basic Algorithm

```
int N = a.Length - 1;
int n = 0;
while (n != N) {
    a[n] = v;
    n = n + 1;
}
```

Example: Decorated

```
Contract.Requires(a.Length > 0);
Contract.Ensures(Contract.ForAll(0, a.Length,
    i => a[i] == v));
int N = a.Length;
int n = 0;
while (n != N) {
    a[n] = v;
    n = n + 1;
}
```

Example: Decorated

```
Contract.Requires(a.Length > 0);
Contract.Ensures(Contract.ForAll(0, a.Length,
    i => a[i] == v));
int N = a.Length;
int n = 0;
Contract.Assert(Contract.ForAll(0, n, i => a[i] == v));
while (n != N) {
    a[n] = v;
    n = n + 1;
    Contract.Assert(Contract.ForAll(0, n,
        i => a[i] == v));
}
```

Example: Termination

Termination function: $T = N - n$
When $n = N$ then $T = 0$.

The End

"Testing shows the presence, not the absence of bugs."
— Edsger W. Dijkstra