Program synthesis

Tallinn University of Technology

A deductive approach to program synthesis by Manna & Waldinger is used

Specification

Specification of a program allows us to express the purpose of the desired program

It does not indicate an algorithm to achieve the purpose

Specification

Typically, specifications involve such constructs as the quantifiers:

for all

for some

the set constructor {x: ...}

the descriptor find z such that . . .

Specification

A program to compute the integer square root of a nonnegative integer n:

sqrt(n) ≔ find z such that

integer(z) and $z^2 \leq n < (z + 1)^2$

where integer(n) and 0 ≤ n.

Specification

A program to compute the integer square root of a nonnegative integer n

sqrt(n) ≔ find z such that

integer(z) and $z^2 \leq n < (z+1)^2$

where integer(n) and 0 ≤ n.

Output condition

Specification

A program to compute the integer square root of a nonnegative integer n

sqrt(n) $:=$ *find z such that*

integer(z) and $z^2 \leq n < (z+1)^2$

where integer(n) and 0 $\leq$ *n*.

Input condition

Specification

A program to sort a list l:

sqrt(l) ≔ find z such that
ordered(z) and perm(l,z)
where islist(l).

Synthesis

General form of specification (Manna and Waldinger)

$f(a) \coloneqq$ *find z such that*

$\quad R(a,z)$

$\quad$ *where* $P(a)$.

a denotes the input and *z* the output of the desired program.

Deductive synthesis

A program is derived from the specification by attempting to prove a theorem of the form:

for all a,

if P(a)

then for some z, R(a, z).

Deductive synthesis

Proof must be constructive – it must tell us how to find an output z satisfying the desired output condition.

A program to compute z can be extracted from the proof.

Sequent

Sequent consists of two list of sentences -

Assertions (A1, A2, ..., Am)

Goals (G1, G2, ..., Gn)

Sequent

Each assertion and goal may be associated with an entry called *output expression*.

Output expression records the program segment that has been constructed at each stage of the *derivation*

Sequent

Assertions	*Goals*	*Output*
A1(a,x)		t1(a,x)
	G1(a,x)	T2(a,x)

Sequent

The sequent for program specification:

f(a) $\coloneqq$ *find z such that*

R(a,z)

where P(a).

Assertions	*Goals*	*Output*
P(a)		
	R(a,z)	z

Sequent

The sequent for program specification:

(f(a),g(a)) ≔ find (y,z) such that
R(a,y,z)
where P(a).

Assertions	*Goals*	*Output*	*Output*
P(a)			
	R(a,y,z)	y	z

Sequent

Meaning of the sequent:

If all instances of each of the assertions are true, then some instances of at least one of the goals is true.

Sequent

*if **for all** x, Al(a, x) **and***
for all x, A2(a, x) and
•
•
•
for all x, Am(a, x)
*then **for some** x, G1(a, x) **or***
for some x, G2(a, x) or
•
•
•
for some x, Gn(a, x)

Deductive system

New assertions and goals, and corresponding new output expressions, are added to the sequent.

The addition must not alter the meaning of the sequent.

Deductive system

The process terminates if the goal *true* (or the assertion *false*) is produced.

The final output expression consists entirely of primitives from the target programming language.

Deductive system

The output expression on termination is the desired program.

	true	t

OR

False		t

$f(a) \coloneqq \mathrm{t}$

t is the desired program.

Rules of deduction

Splitting

Transformation

Resolution

Mathematical induction

Assertions and goals are created and added by application of rules.

Splitting rule

The splitting rules allow us to decompose an assertion or goal into its logical components.

andsplit

orsplit

ifsplit

Splitting rule - andsplit

Assertions	*Goals*	*Output*
F and G		t
F		t
G		t

andsplit – If the sequent contains F **and** G then two assertions F and G are added without changing the meaning of the sequent.

Splitting rule – orsplit

Assertions	*Goals*	*Output*
	F **or** G	t
	F	t
	G	t

orsplit – If the sequent contains goal F **or** G then two goals F and G are added without changing the meaning of the sequent

Splitting rule - ifsplit

Assertions	*Goals*	*Output*
	if F **then** G	t
F		t
	G	t