Program Repair Example

```
int is_upward(int in, int up, int down){
      int bias, r;
 3
      if (in)
        bias = down; // fix: bias = up + 100
 5
      else
6
        bias = up;
7
      if (bias > down)
        r = 1;
9
      else
10
      r = 0;
11
      return r;
12
            Inputs
                              Output
 Test
                                  observed Passed?
                 down | expected
             up
        _{
m in}
                   100
              0
                            0
  2
             11
                   110
  3
            100
                    50
            -20
                    60
  5
                    10
                   -10
```

Program Repair Example

```
int is_upward(int in, int up, int down){
      int bias, r;
 3
      if (in)
                 c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
         bias =
      else
6
         bias = up;
      if (bias > down)
         r = 1;
9
      else
10
         r = 0:
11
      return r;
12
            Inputs
                               Output
 Test
                          expected
                                    observed
                                              Passed?
             up
                  down
        _{
m in}
                    100
              0
                             0
  2
              11
                    110
  3
             100
                     50
             -20
                     60
  5
         0
              0
                     10
                    -10
```

Program Repair Example

```
int is_upward(int in, int up, int down) {
       int bias, r;
 3
       if (in)
                 |c_0| + |c_1| * bias + |c_2| * in + |c_3| * up + |c_4| * down;
         bias =
       else
 6
                                           c0 = 100
         bias = up;
      if (bias > down)
                                           c1 = 0
         r = 1:
                                           c2 = 0
       else
                                           c3 = 1
10
         r = 0:
                                           c4 = 0
11
       return r;
                                            "bias = up +
12
                                           100:"
             Inputs
                                Output
 Test
                          expected
                                     observed Passed?
              up
                   down
        in
                     100
               0
                              0
  2
              11
                     110
  3
             100
                      50
             -20
                      60
  5
         0
               0
                      10
               0
                     -10
```

Reachability Example

```
int x, y; /* global input */
int P()
  if (2 * x == y)
    if (x > y + 10)
  return 0;
```

Reachability Example

```
int x, y; /* global input */
int P()
  if \quad (2 * x == y)
    if (x > y + 10)
  return 0;
```

```
int is_upward(int in, int up, int down) {
 ^{2}
      int bias, r;
      if (in)
 3
        bias = c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
      else
 6
       bias = up;
      if (bias > down)
8
         r = 1;
9
      else
10
         r = 0;
11
      return r;
12
            Inputs
                                Output
 Test
                          expected observed Passed?
                   down
             up
        _{
m in}
                    100
                                        0
             0
  ^{2}
              11
                    110
  3
             100
                     50
         0
             -20
                     60
  5
               0
                     10
  6
                     -10
               0
```

```
int is_upward(int in, int up, int down){
 ^{2}
      int bias, r;
 3
      if (in)
       bias = c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
 5
      else
6
      bias = up;
      if (bias > down)
7
8
      r = 1;
9
      else
10
        r = 0;
11
      return r;
12
           Inputs
                             Output
 Test
                 down expected observed Passed?
            up
       _{
m in}
                  100
                           0
            0
  ^{2}
            11
                  110
  3
            100
                  50
        0
                                     int x, y; /* global input */
            -20
                 60
  5
        0
             0
                   10
                                     int P() {
  6
             0
                   -10
                                        if (2 * x = y)
                                          if (x > y + 10)
                                        return 0;
```

```
int is_upward(int in, int up, int down){
      int bias, r;
 3
      if (in)
       bias = c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
      else
      bias = up;
      if (bias > down)
8
      r = 1;
      else
10
        r = 0;
11
      return r;
12
```

Output

"Heart" insights:

Multiple tests make Synthesis difficult.

Multiple path conditions make Reachability difficult.

```
Test
                 down expected observed Passed?
            up
       _{
m in}
                   100
            0
                            0
            11
                   110
 3
           100
                   50
            -20
                 60
 5
             0
                    10
 6
             0
                   -10
```

Inputs

```
int x, y; /* global input */
int P() {
  if (2 * x == y)
    if (x > y + 10)
    [L]

return 0;
```

```
int is_upward(int in, int up, int down){
 ^{2}
       int bias, r;
 3
       if (in)
        bias = |c_0| + |c_1| * bias + |c_2| * in + |c_3| * up + |c_4| * down;
 5
       else
 6
         bias = up;
                                     int c_0, c_1, c_2, c_3, c_4; /* global input */
 7
       if (bias > down)
                                     int p<sub>is_upward</sub> (int in, int up, int down) {
 8
       r = 1;
                                        int bias, r;
9
       else
                                        if (in)
10
         r = 0;
                                          bias = c_0+c_1*bias+c_2*in+c_3*up+c_4*down;
11
       return r;
                                        else
12
                                          bias = up;
                                        if (bias > down)
             Inputs
                                 Out
                                           r = 1;
 Test
                   down
                           expected
              up
         in
                                        else
                                           r = 0;
               0
                     100
                               0
                                        return r;
  \mathbf{2}
              11
                     110
                               1
          1
   3
             100
                      50
         0
                               1
              -20
                      60
                                     int main() {
                                         if(p_{is\_upward}(1,0,100) == 0 \&\&
   5
                      10
         0
               0
                               0
                                            p_{is\_upward}(1,11,110) == 1 \&\&
  6
               0
                     -10
                                            p_{is\_upward}(0,100,50) == 1 \&\&
                       Conver
                                            p_{is\_upward}(1, -20, 60) == 1 \&\&
                                            p_{is\_upward}(0,0,-10) == 1){
                                            [L]
                                         return 0;
```

```
int is_upward(int in, int up, int down){
    ^{2}
                         int bias, r;
    3
                         if (in)
                               bias = c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
    5
                         else
                                  bias = Lemma 1:
    6
                                                                                                                                   int c_0, c_1, c_2, c_3, c_4; /* global input */
                        \label{eq:pis_upward} \begin{tabular}{ll} \b
   8
                         elsAgree On Variables int bias, r;
   9
10
                                                                                                                                                    bias = c_0+c_1*bias+c_2*in+c_3*up+c_4*down;
11
                         return r;
                                                                                                                                             else
12
                                                                                                                                                     bias = up;
                                                                                                                                             if (bias > down)
                                               Inputs
                                                                                                                      Out
                                                                                                                                                         r = 1;
    Test
                                                                                                expected
                                                                     down
                                                   up
                               _{
m in}
                                                                                                                                             else
                                                                                                                                                         r = 0;
                                                                           100
                                                       0
                                                                                                                                             return r;
          ^{2}
                                                    11
                                                                           110
                                                                     Lemma 2:
          3
                                                100
          4
                                                                               60
                                                                                                                                   int main() {
                                                                                                Coxcesponds_{upward}(1,0,100) = 0 \&\&
          5
                                                                                                                       Tests" pis_upward (1,11,110) == 1 &&
          6
                                                                                                                                                             p_{is\_upward}(0,100,50) == 1 \&\&
                                                                                                                                                             p_{is\_upward}(1, -20, 60) == 1 \&\&
                                                                                                                                                            p_{is\_upward}(0,0,10) = 0 \&\&
                                                                                                                                                             p_{\rm is\_upward} \, (\, 0 \, , 0 \, , -10\, ) \,\, = \,\, 1\, ) \, \{
                                                                                                                                                          [L]
                                                                                                                                                 return 0;
```

Reachability to Synthesis Example

```
int x, y; /* global input */
int P() {
   if (2 * x == y)
      if (x > y + 10)
                                     int is_upward(int in, int up, int down){
                                       int bias, r;
                                       if (in)
                                         bias = c_0 + c_1 *bias + c_2 *in + c_3 *up + c_4 *down;
   return 0;
                                       else
                                        bias = up;
                                       if (bias > down)
                                      r = 1:
                                       else
                                  10
                                         r = 0:
                                  11
                                       return r;
                                  12
                                            Inputs
                                                           Output
                                                 down expected observed
                                                                        Passed?
                                                  100
                                             0
                                                                  0
                                             11
                                                  110
                                            100
                                                  50
                                            -20
                                                   10
                                                                  0
```

-10

Reachability to Synthesis Example

```
int x, y; /* global input */
int P() 
  if (2 * x = y)
                          Convert
    if (x > y + 10)
                                            int q<sub>main</sub>() {
                 int q_P() {
                                             /* Find |x| and |y|
  return 0;
                  if (2*[x] == [y]]
                                                 Equivalently,
                   if (x > y + 10)
                                                     synthesize:
                                               x = c_x
                     /* location of [L]
                                               y = c_u * /
                         in P * /
                                             try {
                     raise REACHED;
                                                qp();
                                             } catch (REACHED) {
                   return 0;
                                                return 1;
                                              return 0;
                  Test suite: Q()=1
```