

# Knowledge Transfer based Many-Objective Approach for Finding Bugs in Multi-path Loops

Han Huang, Stuart D. Semujju, Fangqing Liu, Yi Xiang, Shuling Yang, and Zhifeng Hao

## 1 C# LOOP PROGRAMS AS RUN IN MICROSOFT PEX

Fig. 1 depicts the modified program diamond\_1-1.c. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-acceleration/diamond\\_1-1.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-acceleration/diamond_1-1.c).

Fig. 2 depicts the modified program diamond\_2-2.c. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-acceleration/diamond\\_2-2.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-acceleration/diamond_2-2.c).

```
public class Diamond_1-1
{
    static int y = 1;
    public static void Main( string[] args )
    {
        diamond_1 ( y );
    }

    public static void diamond_1 ( int y ){
        int x = 0;
        while (x < 10) {
            if (y % 2 == 0) {
                x++;
            } else {
                x += 2;
            }
        }

        if(((x % 2) != (y % 2))); // bug 1
    }
}
```

Fig. 1: diamond\_1-1

Fig. 3 depicts the modified program insertion\_sort-1. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/insertion\\_sort-1.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/insertion_sort-1.c).

Fig. 4 depicts the modified program string-1.c. The original program can be found at: <https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/string-1.c>.

Fig. 5 depicts the modified program veris.c\_OpenSER\_cases1\_stripFullBoth\_arr.c.

- H. Huang, S. D. Semujju, F. Liu, Yi. Xiang and S. Yang are with the School of Software Engineering, South China University of Technology, Guangzhou 510006, P.R.China. E-mail: [hhan@scut.edu.cn](mailto:hhan@scut.edu.cn), [stuartsemujju@gmail.com](mailto:stuartsemujju@gmail.com), [564376030@qq.com](mailto:564376030@qq.com), [gzhuxiang\\_yi@163.com](mailto:gzhuxiang_yi@163.com), and [694712017@qq.com](mailto:694712017@qq.com).
- Z. Hao is with the Department of Mathematics, College of Science Shantou University, Guangdong 515063, P.R.China. E-mail: [zhao@fosu.edu.cn](mailto:zhao@fosu.edu.cn)

```
public class Diamond_2.2
{
    static int y = 0;
    public static void Main( string[] args ) {
        diamond_2 ( y );
    }

    public static void diamond_2 (int y){
        int x = 0;
        while (x < 3) {
            if (y % 2 == 0) x += 2;
            else x++;
            if (y % 2 == 0) x += 2;
            else x -= 2;
            if (y % 2 == 0) x += 2;
            else x += 2;
            if (y % 2 == 0) x += 2;
            else x -= 2;
            if (y % 2 == 0) x += 2;
            else x += 2;
            if (y % 2 == 0) x += 2;
            else x -= 4;
            if (y % 2 == 0) x += 2;
            else x += 4;
            if (y % 2 == 0) x += 2;
            else x += 2;
            if (y % 2 == 0) x += 2;
            else x -= 4;
            if (y % 2 == 0) x += 2;
            else x -= 4;
        }
        if((x % 2) != (y % 2)); // bug 1
    }
}
```

Fig. 2: diamond\_2.2

The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/veris.c\\_OpenSER\\_cases1\\_stripFullBoth\\_arr.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/veris.c_OpenSER_cases1_stripFullBoth_arr.c).

Fig. 6 depicts the modified program vogal-1.c. The original program can be found at: <https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/vogal-1.c>.

Fig. 7 depicts the modified program vogal-2.c. The original program can be found at: <https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/vogal-2.c>.

Fig. 8 depicts the modified program invert\_string-1.c. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/invert\\_string-1.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/invert_string-1.c).

```

public class Insertion_sort-1{

    static int[] d = new int[10];
    public static void Main( string[] args ){
        insertion_sort( d );
    }

    public static void insertion_sort (int [] v ){
        int i = 0;
        int j = 0;
        int k = 0;
        int key = 0;
        int SIZE = 10;
        for (j = 1; j< SIZE ;j++) {
            key = v[j];
            i = j - 1;
            while((i>=0) && (v[i]>key)) {
                if (i<2)
                    v[i+1] = v[i];
                i = i - 1;
            }
            v[i+1] = key;
        }

        for (k=1;k<SIZE;k++){
            if((v[k-1] > v[k]));// bug 1
        }
    }
}

```

Fig. 3: insertion\_sort-1

```

public class String-1{
    static char [] a = new char [5];
    static char [] b = new char [5];
    public static void Main(string [] args){
        String_1(a, b);
    }

    public static void String_1(char [] string_A, char [] string_B)
    {
        int bugcount = 0;
        int i = 0;
        int j = 0;
        int nc_A = 0;
        int nc_B = 0;
        int found=0;
        nc_A = 0;
        while(string_A[nc_A]!='\0'){
            nc_A++;
        }
        nc_B = 0;
        while(string_B[nc_B]!='\0'){
            nc_B++;
        }
        i=j=0;
        while((i<nc_A) && (j<nc_B))
        {
            if(string_A[i] == string_B[j])
            {
                i++;
                j++;
            }
            else
            {
                i = i-j+1;
                j = 0;
            }
        }

        if (j>nc_B-1){
            found = -1;
        }

        else {found = 0;}

        if((found != 0));//bug 1
    }
}

```

Fig.4: string-1

```

public class Verisec
{
    static int BASE_SZ = 2;
    static char EOS = '0';
    static int EOF = -1;
    static int ERR = -1;
    static int NEEDLE_SZ = 2;
    static int EXPRESSION_LENGTH = BASE_SZ;
    static int LINE_LENGTH = EXPRESSION_LENGTH + NEEDLE_SZ + 4;
    static char[] A = new char[LINE_LENGTH+1];

    public static void Main( string[] args )
    {
        A[LINE_LENGTH] = EOS;
        verisecl (A);
    }

    public static void verisecl(char [] str ){
        int bugcount = 0;
        int start=0;
        int i=-1;
        int j=-1;
        do {
            i++;
            switch(str[i]) {
                case EOS:
                    while ((str[start] == ' ') || (str[start] == '\t')){
                        start++;
                    }
                    if (str[start] == '"') {
                        start++;
                    }
                    j = i-1;
                    while ((0 < j) && ((str[j] == ' ') || (str[j] == '\t'))))
                    {
                        j--;
                    }
                    if ((0 < j) && (str[j] == '"')) {
                        j--;
                    }

                    if (start<=j) {
                        if(j - start + 1 < 2);// bug 1
                    }
                    start = i+1;
                } while (str[i] != EOS);
            }
        }
    }
}

```

Fig. 5: veris.c\_OpenSER\_cases1\_stripFullBoth\_arr.c

Fig. 9 depicts the modified program sum\_array-1. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/sum\\_array-1.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loops/sum_array-1.c).

Fig. 10 depicts the modified program benchmark05\_conjunctive.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark05\\_conjunctive.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark05_conjunctive.c).

Fig. 11 depicts the modified program benchmark06\_conjunctive.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark06\\_conjunctive.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark06_conjunctive.c).

Fig. 12 depicts the modified program benchmark40\_polynomial.c. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark40\\_polynomial.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark40_polynomial.c).

Fig. 13 depicts the modified program benchmark44\_disjunctive.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark44\\_disjunctive.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark44_disjunctive.c).

```

public class Vogal-1
{
    static char[] a = new char[5];
    public static void Main( string[] args )
    {
        vogal( a );
    }

    static void vogal (char [] input_string ){
        int MAX = 5;
        int bugcount = 0;
        char[] vogal_array = {'a','A','e','E','i','I','o','O','u','U','\0'};
        int i = 0;
        int j = 0;
        int cont = 0;
        int n_caracter = 0;
        while(input_string[n_caracter]!='\0'){
            n_caracter++;
        }
        for(i=0;i<n_caracter;i++){
            for(j=0;j<MAX/2;j++){
                if(input_string[i] == vogal_array[j])
                    cont++;
            }
            i = 0;
            int cont_aux = 0;
            while(input_string[i]!='\0')
            {
                for(j=0;j<MAX/2;j++){
                    {
                        if(input_string[i] == vogal_array[j])
                            cont_aux++;
                    }
                }
                i++;
            }
            if((cont_aux!=cont)); //bug 1
        }
    }
}

```

Fig. 6: vogal-1.c

lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark44\_disjunctive.c.

Fig. 14 depicts the modified program benchmark47\_linear.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark47\\_linear.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark47_linear.c).

Fig. 15 depicts the modified program benchmark53\_polynomial.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark53\\_polynomial.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/loop-zilu/benchmark53_polynomial.c).

Fig. 16 depicts the modified program bresenham-ll\_unwindbound10.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/bresenham-ll\\_unwindbound10.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/bresenham-ll_unwindbound10.c).

Fig. 17 depicts the modified program dijkstra-u\_unwindbound10.c. The original program can be found at: [https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/dijkstra-u\\_unwindbound10.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/dijkstra-u_unwindbound10.c).

Fig. 18 depicts the modified program cohendiv-ll\_unwindbound2.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/cohendiv-ll\\_unwindbound2.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/cohendiv-ll_unwindbound2.c).

Fig. 19 depicts the modified program divbin\_unwindbound5.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/divbin\\_unwindbound5.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/divbin_unwindbound5.c).

```

public class Vogal-2
{
    static char[] a = new char[10];
    static int MAX = 10;
    public static void Main( string[] args )
    {
        vogal2( a );
    }

    static void vogal2 (char [] input_string ){
        input_string[MAX-1] = '\0';
        char [] vetor_vogais={'a','A','e','E','i','I','o','O','u','U','\0'};
        int i = 0;
        int j = 0;
        int cont = 0;
        int n_caracter = 0;
        while(input_string[n_caracter]!='\0'){
            n_caracter++;
        }
        cont = 0;
        for(i=0;i<n_caracter;i++){
            for(j=0;j<8;j++){
                if(input_string[i] == vetor_vogais[j])
                    cont++;
            }
            i=0;
            int cont_aux = 0;
            while(input_string[i]!='\0')
            {
                for(j=0;j<10;j++){
                    {
                        if(input_string[i] == vetor_vogais[j])
                            cont_aux++;
                    }
                }
                i++;
            }
            if((cont_aux!=cont)); //bug 1
        }
    }
}

```

Fig. 7: vogal-2

```

public class Invert_string-1 {
    static int MAX = 10;
    static char[] str1 = new char[10];
    static char[] str2 = new char[10];
    public static void Main(string[] args) {
        String_1( str1, str2 );
    }

    static void String_1(char [] str1, char [] str2) {
        int cont = 0;
        int i = 0;
        int j = 0;
        str1[MAX-1]='\0';
        j = 0;
        for (i = MAX - 1; i >= 0; i--) {
            str2[j] = str1[i];
            j++;
        }
        j = MAX-1;
        for (i=0; i<MAX; i++) {
            if (str1[i] != str2[j]); //bug 1
            j--;
        }
    }
}

```

Fig. 8: invert\_string-1

```

public class Sum_array-1 {
    static int M = 10;
    static int [] A = new int[10];
    static int [] B = new int[10];
    static int [] C = new int[10];

    public static void Main(string[] args) {

        String_l( A , B , C );
    }

    static void String_l(int [] A, int [] B , int [] C) {

        for(int i =0;i< M;i++) {
            C[i]=A[i]+B[i];
        }
        for(int i=0;i<M;i++) {
            if(C[i]!=A[i]-B[i]); //bug 1
        }
    }
}

```

Fig. 9: sum\_array-1

```

public class Benchmark05_conjunctive{
    static int SIZE = 10;
    static int f=0;
    static int g=0;
    public static void Main( string[] args )
    {
        benchmark05( f, g );
    }
    static void benchmark05(int x, int y) {
        int bugcount =0;
        if (!(x>=0 && x<=y && y<SIZE)) {
            while (x< SIZE) {
                x++;
                if (x>y) {
                    y++;
                }
            }
            if(y!=SIZE);//bug 1
        }
    }
}

```

Fig. 10: benchmark05\_conjunctive

lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/divbin\_unwindbound5.c.

Fig. 20 depicts the modified program egcd-ll\_unwindbound2.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/egcd-ll\\_unwindbound2.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/egcd-ll_unwindbound2.c).

Fig. 21 depicts the modified program hard2\_unwindbound10.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/hard2\\_unwindbound10.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/hard2_unwindbound10.c).

Fig. 22 depicts the modified program man-nativ\_unwindbound10.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/mannativ\\_unwindbound10.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/mannativ_unwindbound10.c).

Fig. 23 depicts the modified program prod4br-ll\_unwindbound5.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/prod4br-ll\\_unwindbound5.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/prod4br-ll_unwindbound5.c).

Fig. 24 depicts the modified program prodbin-

```

public class Benchmark06_conjunctive{
    static int SIZE = 10;
    static int f=0;
    static int g=0;
    static int a = 0;
    static int b = 0;
    static int c = 0;
    static int d = 0;
    static int e = 0;
    static int w = 0;

    public static void Main(string[] args )
    {
        benchmark06( a, b,c,d,e );
    }

    static void benchmark06 (int j, int i , int x, int y, int k){
        int bugcount = 0;
        while ( w < SIZE ) {
            if(j==i) {
                x++;
                y--;
            }
            else {
                y++;
                x--;
            }
            j++;
            w++;
        }
        if(!(x+y==k)); // bug 1
    }
}

```

Fig. 11: benchmark06\_conjunctive

```

public class Benchmark40_polynomial {
    static int SIZE = 4;
    static int l = 0;
    static int w = 0;
    public static void Main(string[] args) {
        benchmark40(w, l);
    }
    static void benchmark40(int x, int y) {
        int i = 0;
        int bugcount = 0;
        while (i < SIZE) {
            if (x == 0) {
                if (y > 0) {
                    x++;
                } else {
                    x--;
                }
            }
            if (x > 0) {
                y++;
            }
            else {
                x--;
            }
            i++;
        }
        if (!(x * y >= 0)) ;// bug 1
    }
}

```

Fig. 12: benchmark40\_polynomial

ll\_unwindbound10.c. The original program can be found at:[https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/prod4br-ll\\_unwindbound10.c](https://gitlab.com/sosy-lab/benchmarking/sv-benchmarks/-/blob/main/c/nla-digbench-scaling/prod4br-ll_unwindbound10.c).

---

```

public class Benchmark44_disjunctive {
    static int SIZE = 4;
    static int l = 0;
    static int w = 0;
    public static void Main(string[] args) {
        benchmark44(w, l);
    }
    static void benchmark44 ( int x, int y) {
        int counter = 0;
        int bugcount =0;
        while ( x < y )
        {
            if(counter == 5){
                break;
            }
            if ((x<0 && y<0)) {
                x=x+7; y=y-10;
            }
            if ((x<0 && y>=0)) {
                x=x+7; y=y+3;
            }
            else {
                x=x+10; y=y+3;
            }
            counter++;
        }
        if(!(x >= y && x <= y + 16)); // bug 1
    }
}

```

---

Fig. 13: benchmark44\_disjunctive

---

```

public class Benchmark47_linear {
    static int l = 0;
    static int w = 0;
    public static void Main(string[] args) {
        benchmark47(w, l);
    }
    static void benchmark47 (int x , int y){
        int counter = 0;
        int bugcount = 0;
        while (x < y)
        {
            if (counter == 4){
                break;
            }
            if (x < 0) {
                x = x + 7;
            }
            else {
                x = x + 10;
            }
            if (y < 0){
                y = y - 10;
            }
            else {
                y = y + 3;
            }
            counter++;
        }
        if(!(x >= y && x <= y + 16)); // bug 1
    }
}

```

---

Fig. 14: benchmark47\_linear

---

```

public class Benchmark53_polynomial {
    static int l = 0;
    static int w = 0;
    public static void Main(string[] args) {
        benchmark53(w, l);
    }
    static void benchmark53 (int x , int y){
        int i = 0;
        int bugcount = 0;
        while (i < 4) {
            if(x==0) {
                if (y>0){
                    x++;
                }
                else {
                    x--;
                }
            }
            if(x>0) {
                y++;
            }
            else {
                x--;
            }
            i++;
        }
        if (!(x*y>=0)) ; // bug 1
    }
}

```

---

Fig. 15: benchmark53\_polynomial

---

```

public class Bresenham-ll_unwindbound10{
    static int X = 1;
    static int Y = 1;
    public static void Main(string[] args) {
        bresenham(X,Y);
    }
    static void bresenham(int X, int Y) {
        long x, y, v, xy, yx;
        v = ((long)2 * Y) - X;
        y = 0;
        x = 0;
        int counter = 0;
        int bugcount1 = 0;
        int bugcount2 = 0;
        while(counter < 10) {
            yx = (long)Y * x;
            xy = (long)X * y;
            if( 2*yx - 2*xy - X + (long) 2*Y - v != 0); // bug 1

            if(!(x <= X)) {
                break;
            }
            if(v < 0) {
                v = v + (long)2 * Y;
            }
            else {
                v = v + 2 * ((long)Y - X);
                y++;
            }
            x++;
            counter++;
        }
        xy = (long ) x*y;
        yx = (long ) Y*x;
        if(2*yx - 2*xy - X + (long ) 2*Y - v + 2*y != 0); // bug 2
    }
}

```

---

Fig. 16: bresenham-ll\_unwindbound10

---

```

public class Dijkstra-u_unwindbound10 {
    static int n = 1;
    public static void Main(string[] args) {
        dijkstra( n);
    }
    static void dijkstra(int n) {
        int p = 0;
        int q = 1;
        int r = n;
        int h = 0;
        int count = 0;
        while (count < 10) {
            if (!(q <= n)){
                break;
            }
            q = 4 * q;
            count++;
        }
        int i = 0;
        while (i < 10) {
            if(r >= 2 * p + q) ;//bug 1
            if(p*p + r*q != n*q);//bug 2

            if(h * h * h - 12 * h * n * q + 16 * n * p * q - h * q * q
            - 4 * p * q * q + 12 * h * q * r - 16 * p * q * r != 0);
                //bug 3

            if(h * h * n - 4 * h * n * p + 4 * (n * n) * q - n * q * q
            - h * h * r + 4 * h * p * r - 8 * n * q * r + q * q * r + 4
            * q * r * r != 0);
                //bug 4
            if(h * h * p - 4 * h * n * q + 4 * n * p * q - p * q * q +
            4 * h * q * r - 4 * p * q * r != 0);
                //bug 5
            if(p * p - n * q + q * r != 0);
                //bug 6
            if (!(q != 1)){
                break;}
            q = q / 4;
            h = p + q;
            p = p / 2;
            if (r >= h) {
                p = p + q;
                r = r - h;
            }
            i++;
        }
        if(h*h*h - 12*h*n + 16*n*p + 12*h*r - 16*p*r - h - 4*p != 0)
            ;//bug
            7
        if(p*p - n + r != 0); //bug 8;
        if(h*h*p - 4*h*n + 4*n*p + 4*h*r - 4*p*r - p != 0);//bug 9
    }
}

```

---

Fig. 17: dijkstra-u\_unwindbound10

---

```

public class Cohendiv-ll_unwindbound2{
    static int x = 0;
    static int y = 0;
    static int q = 0;
    static int r = 0;
    static int a = 0;
    static int b = 0;
    public static void Main(string[] args) {
        cohendiv(x, y);
    }
    static void cohendiv(int x, int y) {
        if(y >= 1) {
            q = 0;
            r = x;
            a = 0;
            b = 0;
            int counter = 0;
            int bugcount = 0;
            while (counter < 2) {
                if(b == y*a);
                if(x == q*y + r);
                if (!(r >= y))
                    break;
                a = 1;
                b = y;
                int i = 0;
                while (i<2) {
                    if(b != y*a) ;//bug 1
                    if (x != q*y + r);//bug 2;
                    if (r < 0); // bug 3

                    if (!(r < 2 * b)){
                        break;}

                    if(r < 2 * y * a);// bug 4
                    a = 2 * a;
                    b = 2 * b;
                    i++;
                }
                r = r - b;
                q = q + a;
                counter++;
            }
            if(x != q*y + r) ;// bug 5
        }
    }
}

```

---

Fig. 18: cohendiv-ll\_unwindbound2

---

```

public class Divbin_unwindbound5{
    static int A = 0;
    static int B = 0;
    public static void Main(string[] args) {

        divbin( A , B);
    }
    static void divbin(int A , int B) {
        if (( B < 2147483647) &&( B >=1 )){
            int q = 0;
            int r = 0;
            int b = 0;
            r = A;
            b = B;
            int i = 0;
            while (i++ <5) {
                if (!(r >= b)) break;
                b = 2 * b;
            }
            int counter = 0;
            while (counter++ <5) {
                if(A != q * b + r){//bug 1
                    if (!(b != B)) break;
                    q = 2 * q;
                    b = b / 2;
                    if (r >= b) {
                        q = q + 1;
                        r = r - b;
                    }
                }
            }
            if(A != q * b + r) ;// bug2
        }
    }
}

```

---

Fig. 19: divbin\_unwindbound5

---

```

public class Egcd-ll_unwindbound2 {
    static int x = 0;
    static int y = 0;
    static int q = 0;
    static int b = 0;
    static int p = 0;
    static int r = 0;
    static int s = 0;
    static int a = 0;
    public static void Main(string[] args) {
        egcd(x, y);
    }
    static void egcd(int x, int y) {
        if ( (x >= 1) && (y >= 1)) {
            a = x;
            b = y;
            p = 1;
            q = 0;
            r = 0;
            s = 1;
            int counter = 0;
            while (counter < 2) {
                if(1 != p * s - r * q){//bug 1
                    if(a != y * r + x * p){//bug 2
                        if(b != x * q + y * s){// bug 3
                            if (!(a != b)){
                                break;
                            }
                        }
                    }
                    if (a > b) {
                        a = a - b;
                        p = p - q;
                        r = r - s;
                    } else {
                        b = b - a;
                        q = q - p;
                        s = s - r;
                    }
                }
                counter++;
            }
            if(a - b != 0){//bug 4
                if(p*x + r*y - b != 0); // bug 5
                if(q*r - p*s + 1 != 0); // bug 6
                if(q*x + s*y - b != 0); // bug 7
            }
        }
    }
}

```

---

Fig. 20: egcd-ll\_unwindbound2

```

public class Hard-ll_unwindbound10 {
    static int A = 0; // to be found by DynaMOSA/ WSA
    static int B = 0; // to be found by DynaMOSA/ WSA
    public static void Main(string[] args) {
        hardII(A, B);
    }
    static void prodbin(int A, int B) {
        int r = A;
        int d = B;
        int p = 1;
        int q = 0;
        int counter = 0;
        while (counter < 10) {
            if(q != 0); //bug 1
            if(r != A); //bug 2
            if(d != B * p); //bug 3
            d = 2 * d;
            p = 2 * p;
            counter++;
        }
        int i = 0;
        while ( i < 10) {
            if(A != q*B + r); //bug 4
            if(d != B*p); //bug 5
            d = d / 2;
            p = p / 2;
            if (r >= d) {
                r = r - d;
                q = q + p;
            }

            i++;
        }
        if(A != d*q + r); //bug 6
        if(B != d); //bug 7
    }
}

```

Fig. 21:hard-ll\_unwindbound10

```

public class Mannadiv_unwindbound10{
    static int x1 = 0;
    static int x2 = 0;
    static int y1 = 0;
    static int y2 = 0;
    static int y3 = 0;
    public static void Main(string[] args) {
        mannadiv(x1, x2);
    }
    static void mannadiv(int x1, int x2) {
        y1 = 0;
        y2 = 0;
        y3 = x1;
        int i = 0;
        int bugcount1 = 0;
        int bugcount2 = 0;
        if ( (x1 >= 0) && (x2!= 0)) {
            while (i < 10) {
                if(y1*x2 + y2 + y3 != x1); // bug 1
                if (!(y3 != 0)){ break;}
                if (y2 + 1 == x2) {
                    y1 = y1 + 1;
                    y2 = 0;
                    y3 = y3 - 1;
                } else {
                    y2 = y2 + 1;
                    y3 = y3 - 1;
                }
                i++;
            }
            if(y1*x2 + y2 != x1); // bug 2
        }
    }
}

```

Fig. 22:mannadiv\_unwindbound10

```

public class Prod4br-ll_unwindbound5 {

    static int f = 0;
    static int g = 0;
    public static void Main(string[] args) {
        prod4b(f, g);
    }

    static void prodbin2(int f, int g) {
        int a = f;
        int b = g;
        int p = 1;
        int q = 0;
        int i = 0;
        while (i < 5) {
            if (q + a * b * p != (long)f * g); //bug 1
            if (!(a!=0 && b!=0)) {
                break;
            }
            if (a % 2 == 0 && b % 2 == 0) {
                a = a / 2;
                b = b / 2;
                p = 4 * p;
            } else if (a % 2 == 1 && b % 2 == 0) {
                a = a - 1;
                q = q + b * p;
            } else if (a % 2 == 0 && b % 2 == 1) {
                b = b - 1;
                q = q + a * p;
            } else {
                a = a - 1;
                b = b - 1;
                q = q + (a + b + 1) * p;
            }
            i++;
        }
        if (q != (long)f * g); // bug 2
        if (a * b != 0); // bug 3
    }
}

```

Fig. 23:prod4br-ll\_unwindbound5

```

public class Prodbin-ll_unwindbound10
{
    static int a = 0;
    static int b = 0;
    static int x = 0;
    static int y = 0;
    static int z = 0;
    public static void Main( string[] args )
    {
        prodbin( a , b );
    }
    static void prodbin (int a, int b ){
        x = a;
        y = b ;
        z = 0;
        int i = 0;
        int bugcount1 = 0;
        int bugcount2 = 0;
        while (i < 10) {
            if(z + x * y != a * b); // bug 1
            if(! (y != 0)) {
                break;
            }
            if (y % 2 == 1) {
                z = z + x;
                y = y - 1;
            }
            x = 2 * x;
            y = y / 2;

            i++;
        }
        if((z != a * b)); // bug 2
    }
}

```

Fig. 24:prodbin-ll\_unwindbound10