Tutorial on Formal Verification of Programs (Part II)

Wenhui Zhang

State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences

P.O.Box 8718, Beijing 100190, China

2012-12-19

This part argues that a combination of model checking and deductive proofs is useful for certifying the correctness of programs.

1. Introduction

Suppose that we are given the following program (the correct one), where the functions in(), isr(), and isk() are specified in Appendix A.

```
#include <stdio.h>
/*****************************************************/
int in();
int isr(int x,int k);
int isk(int n,int k);
/*****************************************************/
int main()
{
    int n=0;
    int m=0;
    int k=1;

    printf("INFO:   system is now active\n");
    while (1) {
        n=in();
        k=isk(n,k);
        m=isr(n,k);
        printf("RESULT: %i\n\n",m);
    }
}
/*****************************************************/
```

Suppose that the correctness claim of the program is as follows.

```
(at line 18): ((m*m)<=n)&&((m*m)+2*m+1>n)
```

Let isr2.c be the file that contains the program (including the functions specified in Appendix A) and isr.sp be the file that contains the correctness claim as specified above.

The command lines for checking the program is as follows.

```
./c2vvm isr2.c isr.sp
./verds isr2.vvm
```

The model checking time is 1233 seconds.

2. Deductive Proofs (DP)

We take the function “in()” as an example. The function is as follows.

```
int in()
{
        char c=0;
        int k=0;

        while (1) {
                k=0;
                putc('N',stdout); putc(':',stdout); putc(9,stdout);
                c=getc(stdin);
                if (c=='\n') {
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                if (c<'0'||c>'9') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                k=c-'0';
                c=getc(stdin);
                if (c=='\n') { return k; }
                if (c<'0'||c>'9') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
```

```
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                if (k<2) k=k*10+(c-'0');
                else if (k==2&&c=='0') k=20;
                else {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                   }
                   printf("INFO:   the input number must be in {0,...,20}\n\n ");
                   continue;
                }
                c=getc(stdin);
                if (c!='\n') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                return k;
        }
}
```

That we want to prove is the following.

The result of “r=in()” satisfies the formula “r>=0&&r<=20”

We prove this claims as follows (informally, by arguing that the assertions in the different places hold).

```
int in()
{
        char c=0;
        int k=0;
        {}
        while (1) { {}
                k=0;
                putc('N',stdout); putc(':',stdout); putc(9,stdout);
                c=getc(stdin);
                if (c=='\n') {
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
```

```
                        continue;
                } {}
                if (c<'0'||c>'9') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                {c>= '0'&&(c-'0')<=20} k=c-'0'; {k>=0&&k<=20}
                c=getc(stdin); {k>=0&&k<=20}{k>=0}
                if (c=='\n') { {k>=0&&k<=20} return k; } {k>=0}
                if (c<'0'||c>'9') {}{
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                } { k>=0&&c>= '0'&&c<=10+ '0'}
                if (k<2) { k*10+(c-'0')>=0&&k*10+(c-'0')<=20} k=k*10+(c-'0');
                else if (k==2&&c=='0') {} k=20;
                else {}{
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                   }
                   printf("INFO:   the input number must be in {0,...,20}\n\n ");
                   continue;
                }
                {k>=0&&k<=20} c=getc(stdin);
                if (c!='\n') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                {k>=0&&k<=20} return k;
        }
{ for all v, (v>=0&&v<=20) } }
```

This shows the process of proving the claim. For a formal proof, we need a set of rules and formal semantics of the statements involved in the function.

Suppose that the proof has been done without problems. This means that we have proved that the following is correct.

FUNCTION	r=in()
ASSUMPTION (on initial values of variables)	TRUE
GUARANTEE (on final values of variables)	r>=0&&r<=20

2.1. Combining DP with Model Checking

Let the file "isr.fsp" contain the following information.

```
FUNCTION      r=in()
ASSUMPTION    TRUE;
GUARANTEE     r>=0&&r<=20;
```

We verify the program again with this information, as follows.

```
./c2vvm isr2.c isr.sp isr.fsp
./verds isr2.vvm
```

Then a selected part of the output (of which the full one is presented in Appendix B) of the second command is as follows.

```
VERSION:    verds 1.42 - SEP 2012
FILE:       isr2.vvm
PROPERTY:   A G (! (pc = 4 )| (((m * m ){ n )& (((m * m )+ ((2 * m )+ 1 ))> n )))
bound =  0  time = 113
----------  time = 113
bound =  1  time = 113
.
.
bound =104  time = 205
----------  time = 205
CONCLUSION: TRUE (time=205)
```

The following table compares the model checking times of these two verification approaches.

	Time	Time (with A/G)
Model Checking	1233	205

Clearly, such a combination speeds up the model checking time and makes the approach scalable to larger programs.

3. Compositional Model Checking

Note that whether the function "in()" satisfies the assumption-guarantee specification can also be proved by model checking, by constructing a program for this function.

The following is the experimental result of checking the assumption-guarantee specification of the function "in()".

```
./verds -ck in isr2.vvm
VERSION:    verds 1.42 - SEP 2012
FILE:       isr2.vvm
bound =  0  time = 110
----------  time = 110
bound =  1  time = 110
----------  time = 110
INFO:       A/G=1
CONCLUSION: TRUE (time=111)
```

Then we have the following table for comparison.

	Time	Time (with A/G)	Time (A/G)
Model Checking	1233	205	111

This means that we may split a model checking task into two or more model-checking subtasks for improving the scalability of the model checking approach.

A. Appendix

The following are the three functions involved in the main program.

```
int isr(int x,int k)
{
        int y1=0;
        int y2=0;
        int y3=0;

        y1=0;
        y2=1;
```

```
        y3=1;
        if (x==2||(x>2&&k==20)) x=x-1;
        while (y3<=x) {
                y1=y1+1;
                y2=y2+2;
                y3=y3+y2;
        }
        return y1;
}
/*************************************************/
int isk(int n,int k)
{
        if (k!=20) {
                if (k!=n) k=21; else if (k==19) k=0; else k=k+2;
        } else {
                k=21;
        }
        return k;
}
/*************************************************/
int in()
{
        char c=0;
        int k=0;

        while (1) {
                k=0;
                putc('N',stdout); putc(':',stdout); putc(9,stdout);
                c=getc(stdin);
                if (c=='\n') {
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                if (c<'0'||c>'9') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                k=c-'0';
                c=getc(stdin);
                if (c=='\n') { return k; }
```

```
                if (c<'0'||c>'9') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                if (k<2) k=k*10+(c-'0');
                else if (k==2&&c=='0') k=20;
                else {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                   }
                   printf("INFO:   the input number must be in {0,...,20}\n\n ");
                   continue;
                }
                c=getc(stdin);
                if (c!='\n') {
                        while (1) {
                                c=getc(stdin);
                                if (c=='\n') break;
                        }
                        printf("INFO:   the input must be 1 or 2 digits\n\n");
                        continue;
                }
                return k;
        }
}
```

B. Appendix

The full output of the command "./verds isr2.vvm" is as follows.

```
VERSION:    verds 1.42 - SEP 2012
FILE:       isr2.vvm
PROPERTY:   A G (! (pc = 4 )| (((m * m ){ n )& (((m * m )+ ((2 * m )+ 1 ))> n )))
bound =  0  time = 113
----------  time = 113
bound =  1  time = 113
----------  time = 113
bound =  2  time = 113
```

```
----------  time = 113
bound =  3  time = 113
----------  time = 113
bound =  4  time = 114
----------  time = 114
bound =  5  time = 117
----------  time = 117
bound =  6  time = 117
----------  time = 117
bound =  7  time = 117
----------  time = 117
bound =  8  time = 117
----------  time = 117
bound =  9  time = 118
----------  time = 118
bound = 10  time = 119
----------  time = 119
bound = 11  time = 119
----------  time = 119
bound = 12  time = 119
----------  time = 119
bound = 13  time = 119
----------  time = 119
bound = 14  time = 121
----------  time = 121
bound = 15  time = 122
----------  time = 122
bound = 16  time = 122
----------  time = 122
bound = 17  time = 122
----------  time = 122
bound = 18  time = 122
----------  time = 122
bound = 19  time = 124
----------  time = 124
bound = 20  time = 126
----------  time = 126
bound = 21  time = 126
----------  time = 126
bound = 22  time = 126
----------  time = 126
bound = 23  time = 126
----------  time = 126
bound = 24  time = 127
```

```
----------  time = 127
bound = 25  time = 129
----------  time = 129
bound = 26  time = 129
----------  time = 129
bound = 27  time = 129
----------  time = 129
bound = 28  time = 129
----------  time = 129
bound = 29  time = 131
----------  time = 131
bound = 30  time = 132
----------  time = 132
bound = 31  time = 132
----------  time = 132
bound = 32  time = 132
----------  time = 132
bound = 33  time = 133
----------  time = 133
bound = 34  time = 134
----------  time = 134
bound = 35  time = 136
----------  time = 136
bound = 36  time = 136
----------  time = 136
bound = 37  time = 136
----------  time = 136
bound = 38  time = 136
----------  time = 136
bound = 39  time = 139
----------  time = 139
bound = 40  time = 141
----------  time = 141
bound = 41  time = 141
----------  time = 141
bound = 42  time = 141
----------  time = 141
bound = 43  time = 141
----------  time = 141
bound = 44  time = 143
----------  time = 143
bound = 45  time = 145
----------  time = 145
bound = 46  time = 145
```

```
----------  time = 145
bound = 47  time = 145
----------  time = 145
bound = 48  time = 145
----------  time = 145
bound = 49  time = 147
----------  time = 147
bound = 50  time = 149
----------  time = 149
bound = 51  time = 149
----------  time = 149
bound = 52  time = 149
----------  time = 149
bound = 53  time = 149
----------  time = 149
bound = 54  time = 151
----------  time = 151
bound = 55  time = 154
----------  time = 154
bound = 56  time = 154
----------  time = 154
bound = 57  time = 154
----------  time = 154
bound = 58  time = 154
----------  time = 154
bound = 59  time = 156
----------  time = 156
bound = 60  time = 158
----------  time = 158
bound = 61  time = 158
----------  time = 158
bound = 62  time = 158
----------  time = 158
bound = 63  time = 158
----------  time = 158
bound = 64  time = 161
----------  time = 161
bound = 65  time = 163
----------  time = 163
bound = 66  time = 163
----------  time = 163
bound = 67  time = 163
----------  time = 163
bound = 68  time = 163
```

```
----------  time = 163
bound = 69  time = 165
----------  time = 165
bound = 70  time = 168
----------  time = 168
bound = 71  time = 168
----------  time = 168
bound = 72  time = 168
----------  time = 168
bound = 73  time = 168
----------  time = 168
bound = 74  time = 171
----------  time = 171
bound = 75  time = 173
----------  time = 173
bound = 76  time = 173
----------  time = 173
bound = 77  time = 173
----------  time = 173
bound = 78  time = 173
----------  time = 173
bound = 79  time = 176
----------  time = 176
bound = 80  time = 179
----------  time = 179
bound = 81  time = 179
----------  time = 179
bound = 82  time = 179
----------  time = 179
bound = 83  time = 179
----------  time = 179
bound = 84  time = 181
----------  time = 181
bound = 85  time = 184
----------  time = 184
bound = 86  time = 184
----------  time = 184
bound = 87  time = 184
----------  time = 184
bound = 88  time = 184
----------  time = 184
bound = 89  time = 187
----------  time = 187
bound = 90  time = 190
```

```
----------  time = 190
bound = 91  time = 190
----------  time = 190
bound = 92  time = 190
----------  time = 190
bound = 93  time = 190
----------  time = 190
bound = 94  time = 193
----------  time = 193
bound = 95  time = 196
----------  time = 196
bound = 96  time = 196
----------  time = 196
bound = 97  time = 196
----------  time = 196
bound = 98  time = 196
----------  time = 196
bound = 99  time = 199
----------  time = 199
bound =100  time = 202
----------  time = 202
bound =101  time = 202
----------  time = 202
bound =102  time = 202
----------  time = 202
bound =103  time = 202
----------  time = 202
bound =104  time = 205
----------  time = 205
CONCLUSION: TRUE (time=205)
```