Interoperability between R and OpenModelica

Prof. Kannan Moudgalya, Indian Institute of Technology, Bombay
Digvijay Singh, Indian Institute of Technology, Bombay
Arunkumar Palanisamy, RISE, Sweden

January 31, 2022
Introduction to the R Programming Language
Objective and Approach
System Specifications
Example: General-purpose Optimization
Operations performed
Results
Code on GitHub
Limitations of current approach
Future Work
Introduction to the R Programming Language

- R is a programming language for statistical computing and graphics.
- R provides a wide variety of statistical and graphical techniques and is highly extensible.
- R currently provides access to 18830 external packages.
- Practical applications in business, drug advancement, finance, health care, marketing, medicine, etc.

Figure 1: R logo
Objective and Approach

- Call C from OpenModelica and pass parameter values to it.
- Call R from C and pass parameter values to R.
- Print results obtained from R and read them as a single string in C.
- Segregate the string into an array and convert every element to a floating-point number.
- Pass those floating-point numbers to OpenModelica.
Objective and Approach Contd...

OpenModelica

Figure 2: Procedure implemented for interoperability
Interoperability was implemented on the following -

- Windows 10 (64-bit operating system)
- R 3.6.3 (64-bit)
- Open-Modelica v1.16.0-dev-371-geb234c072 (64-bit)
Example: General-purpose Optimization

Figure 3: General-purpose optimization in R using optim() function
Operations performed

- **Running external C file**

```c
external "C" annotation(Library={"Interoperate", "Function.dll","Gradient.dll"}, LibraryDirectory="modelica://R_OM");
```

**Code 1: Running external C file**

- **Calling R from C**

```c
char cmd[1000]="";
strcat(cmd, "Rscript OMR.R ");
char str1[100];
sprintf(str1, "%g", initial_par); // Store the integer value
"initial_par" as a string
strcat(cmd, str1);
strcat(cmd, " ");
```

**Code 2: Calling R from C**
Operations performed Contd...

- **Printing results from R on console**

  ```r
  cat(res_par, res_value, res_fn_counts, res_gr_counts, res_convergence)
  ```

  Code 3: Printing results from R on console

- **Obtaining results in C**

  ```c
  // Create a buffer to read output from console
  int buffersize = 100000;
  char buf[buffersize];
  FILE *fp;
  if ((fp = popen(cmd, "r")) == NULL) {
    printf("Error while opening pipe!\n");
  }
  while (fgets(buf, buffersize, fp) != NULL) {
  ```
Operations performed Contd...

```c
// printf("\nOutput value : %s", buf);
char str[strlen(buf)];
strcpy(str, buf);
// printf("\n Copied Output value : %s", str);
// Split the string and store it in a character array
char *p = strtok(str, " ");
char *array[5];
int i = 0;
while (p != NULL)
{
    array[i++] = p;
    p = strtok(NULL, " ");
}
```
```c
for (int i = 0; i < 5; ++i)
{
  // printf("%s\n", array[i]);
  // Copy the output
  output[i] = atof(array[i]);
  // printf("%f\n", output[i]);
}
return 0;
}
pclose(fp);
```

Code 4: Obtaining results in C
Figure 4: Results obtained after simulation
Code on GitHub

- **Link:** https://github.com/chrl3hr5/OMR

![GitHub Repository](image)

**Figure 5:** Download complete code from GitHub
Limitations of current approach

Following are the limitations of the current approach -

- User must have some knowledge of both R and C.
- It is required to know the number of expected outputs from R beforehand.
- Depending upon the input parameter values, it could be required to make changes in both the C and R scripts for the program to work.
Future work involves the utilization of R API as an alternative approach for R & OpenModelica interoperability and implementation of the same over the Linux operating system.

**Figure 6: R API**

**Source:** https://cran.r-project.org/doc/manuals/r-release/R-exts.html#The-R-API