Performance Analysis for R: Towards a Faster R Interpreter

Helena Kotthaus

joint work with: I. Korb, M. Künne, P. Marwedel
SFB876: Providing Information by Resource-Constrained Data-Analysis

Project A3: Methods for Efficient Resource Utilization in Machine Learning Algorithms

→ Cooperation between statistics and computer science departments at TU Dortmund University

Challenges:
Analysis of high-dimensional genomic data, e.g. survival time analysis
→ unacceptably slow execution of computation-intensive R programs

Goal:
Reduce resource consumption of statistical learning algorithms with a new compiler strategy
Outline

- Performance Analyses
- TraceR – R Profiling Tool
- Runtime and Memory Profiles
- Future Work
Runtime and Memory Consumption Analyses for R Programs

Goals:
- Uncover bottlenecks of real-world R code
- Support development of alternative R interpreters by providing optimization ideas

Bottleneck Analysis:
- Machine learning algorithms
- Real world input data sets from UCI
- Profiling with our TraceR tool

Analysis of:
- Runtime behavior
- Memory consumption

References:

Profiling – TraceR

- Deterministic profiling for the R Language
- Collects information about runtime and memory behavior

- Originally developed for R V. 2 at Purdue University
- New Version for R V. 3 developed by TU Dortmund
  - Added profiling for vector data structures
  - Added dynamic memory profiles and call graph generation
  - Improved usability for R users

- Download & Install
  → `git clone git@github.com:allr/traceR-installer.git`
  `make PREFIX=$HOME/install-tracer`
Runtime Profiling – TraceR vs. Rprof

Example:

Three User Functions

```r
XRES = 1000
YRES = 1000

# quantize the result values to true/false values
quant <- function(v) {
  return(Re(v*v) < 16)
}

# calculate the c constant for each pixel
calcC <- function(imag, real) {
  return((-1 + 2 * imag) * 1i + (-2.3 + 3.0*real))
}

# calculate a mandelbrot fractal the vectorized way
calcMandel <- function() {
  image <- matrix(0+0i, ncol=XRES, nrow=YRES)
  pixelc <- outer((1:YRES)/YRES, (1:XRES)/XRES, calcC)
  for (i in 1:200) {
    image <- image * image + pixelc
  }
  quant(image)
}

image(calcMandel())
```
Runtime Profiling – TraceR vs. Rprof

Rprof Output:
- Function calcC is missing
- Running the profiler multiple times changes the list of functions
Runtime Profiling – TraceR vs. Rprof

TraceR Output: `Rscript --time=demo.time --timeR-quiet demo.R`

<table>
<thead>
<tr>
<th>name</th>
<th>self_time</th>
<th>total_time</th>
<th>number_of_calls</th>
<th>error_exits</th>
<th>is_compiled</th>
</tr>
</thead>
<tbody>
<tr>
<td>demo.R:quant</td>
<td>1684</td>
<td>78523215</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>demo.R:calcC</td>
<td>1952</td>
<td>310858304</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>demo.R:calcMandel</td>
<td>454</td>
<td>9090805841</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- All functions are now present
- Running TraceR multiple times does not change the list
- Disadvantage → Timing overhead and portability
Runtime Behavior Analyses for R

- 30% of the total runtime is spent in *builtin-functions* that contain *type checks and conversion*.
- Up to 17% of the total runtime is spent in *looking up variables & functions*.

![Runtime Behavior Analyses Diagram](image-url)
Memory Consumption Analyses for R

- 44% of allocated memory used for interpreter internal data structures
- 23% of the runtime is spent in memory management
- 58% of all vectors allocated are single-element vectors
- Vector representation requires 10 times more memory as the mere scalar data
Memory-over-Time Profile

- Indicates if your program has a memory leak
- Denotes how much main memory is needed to run your program without page I/Os
Dynamic Page Sharing Optimization for R

Memory-over-time profile with page sharing → memory reduction by 53%

Dynamic Page Sharing Optimization for the R Language H. Kotthaus, I. Korb, M. Engel, P. Marwedel, submitted to Dynamic Languages Symposium

→ Page sharing optimization to reduce memory consumption of large data structures
→ For lssvm page I/Os were reduced which results in a runtime speed up of 5x
Summary & Future Work

- **TraceR** – goal:
  - Uncover bottlenecks of R Programs and support the development of R interpreters

- **Download & Install:**
  - [https://github.com/allr/traceR](https://github.com/allr/traceR)

- **Benchmarks:**
  - [https://github.com/allr/benchR](https://github.com/allr/benchR)

- **Long-term goal:** resource efficient parallel R
  - Enables larger problem sizes