Performance Analysis for R : Towards a Faster R Interpreter

Helena Kotthaus
joint work with: I. Korb, M. Künne, P. Marwedel

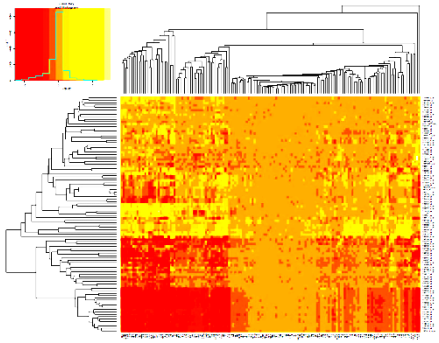
technische universität dortmund

SFB 876 Providing Information by Resource-Constrained Data Analysis

06/26/2014

TU Dortmund
Collaborative Research Center

- 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

Runtime and Memory Consumption Analyses for Machine Learning R Programs, H. Kotthaus, I. Korb, M. Lang, B. Bischl, J. Rahnenführer, P. Marwedel, In Journal of Statistical Computation an Simulation

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

```
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("demo-rp.out"); source("demo.R"); Rprof(NULL);
    summaryRprof("demo-rp.out")
$by.self
                 self.time self.pct total.time total.pct
"*"                    6.50    63.73
"+"                    2.86    28.04
"image.default"        0.72     7.06
"diff"                 0.04     0.39
"calcMandel"           0.02     0.20
"<"                    0.02     0.20
"matrix"               0.02     0.20
"seq.default"          0.02     0.20
```

```
$by.total
                 total.time total.pct self.time self.pct
"eval"                10.20    100.00      0.00     0.00
"image"               10.20    100.00      0.00     0.00
"source"              10.20    100.00      0.00     0.00
"withVisible"         10.20    100.00      0.00     0.00
"calcMandel"           9.42     92.35      0.02     0.20
"*"                    6.50     63.73      6.50    63.73
"+"                    2.86     28.04      2.86    28.04
"image.default"        0.78      7.65      0.72     7.06
"FUN"                  0.16      1.57      0.00     0.00
"outer"                0.16      1.57      0.00     0.00
"diff"                 0.04      0.39      0.04     0.39
"quant"                0.04      0.39      0.00     0.00
"<"                    0.02      0.20      0.02     0.20
"matrix"               0.02      0.20      0.02     0.20
"seq.default"          0.02      0.20      0.02     0.20
"seq"                  0.02      0.20      0.00     0.00
$sample.interval
[1] 0.02
```

Rprof Output:

→Function calcC is missing

→Running the profiler multiple times changes the list of functions

Runtime Profiling – TraceR vs. Rprof

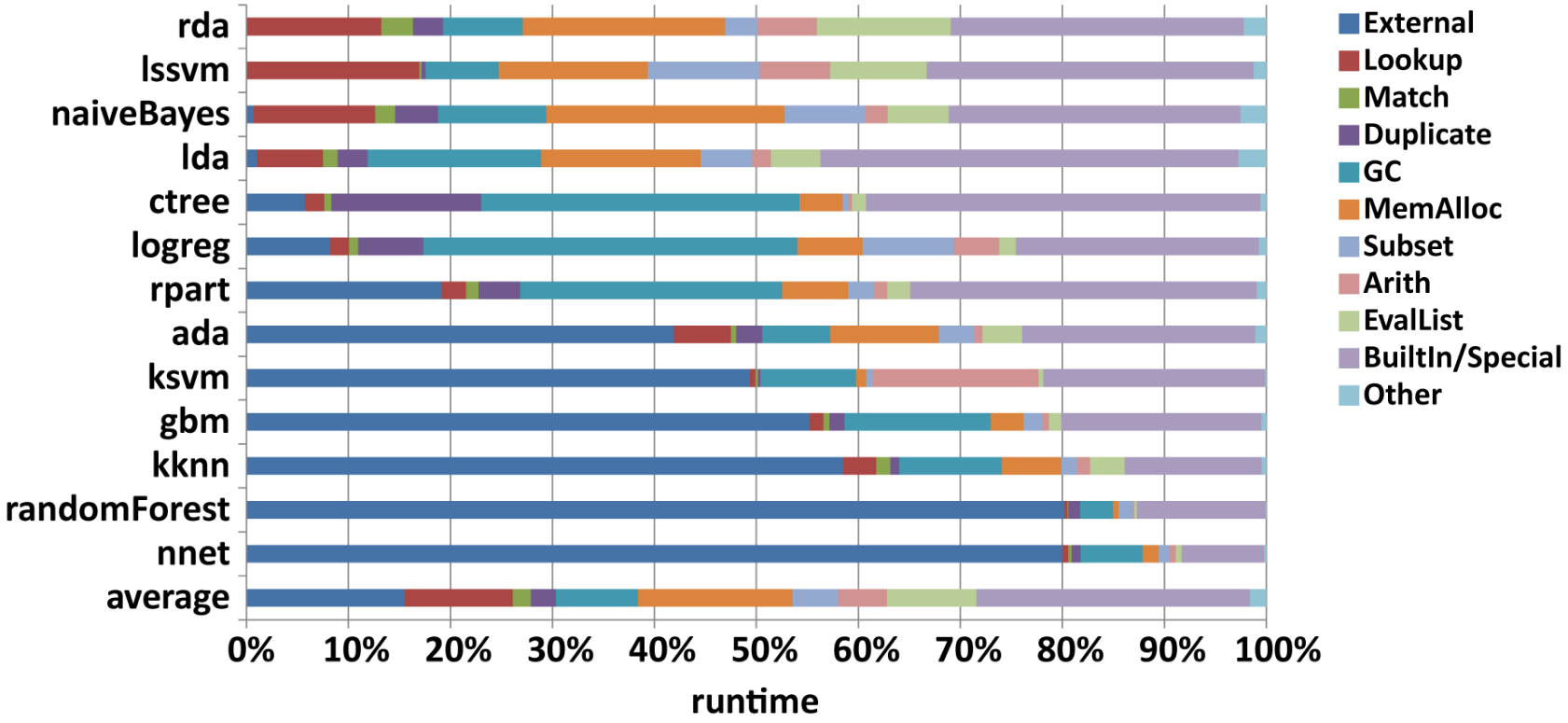
TraceR Output: `Rscript --time=demo.time --timeR-quiet demo.R`

name	self_time	total_time	number_of_calls	error_exits	is_compiled
demo.R quant	1684	78523215	1	0	0
demo.R calcC	1952	310858304	1	0	0
demo.R calcMandel	454	9090805841	1	0	0

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

Runtime Behavior Analyses for R

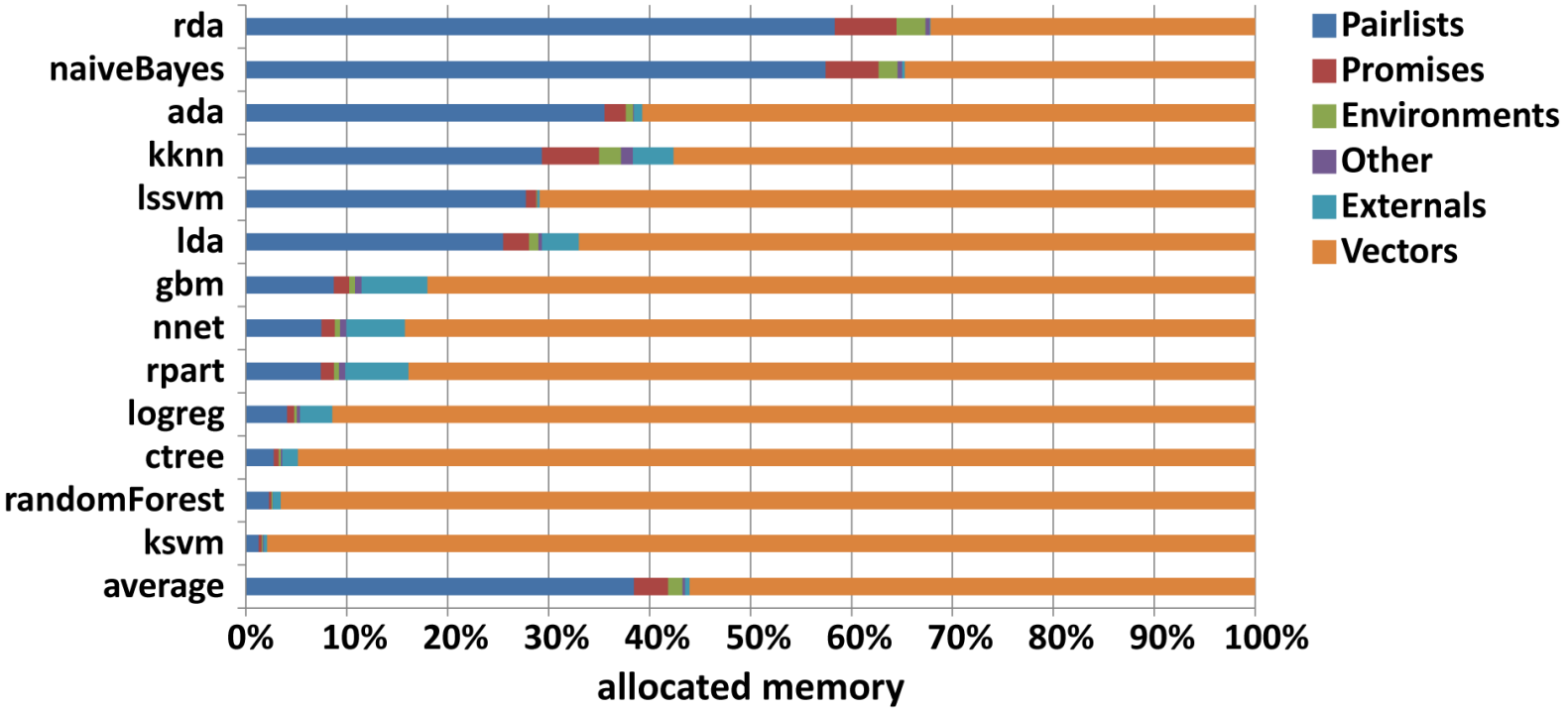
- *30%* of the total runtime is spent in *builtin-functions* that contain *type checks and conversion*
- Up to *17%* of the total runtime is spent in *looking up variables & functions*



Helena Kotthaus
Computer Science XII

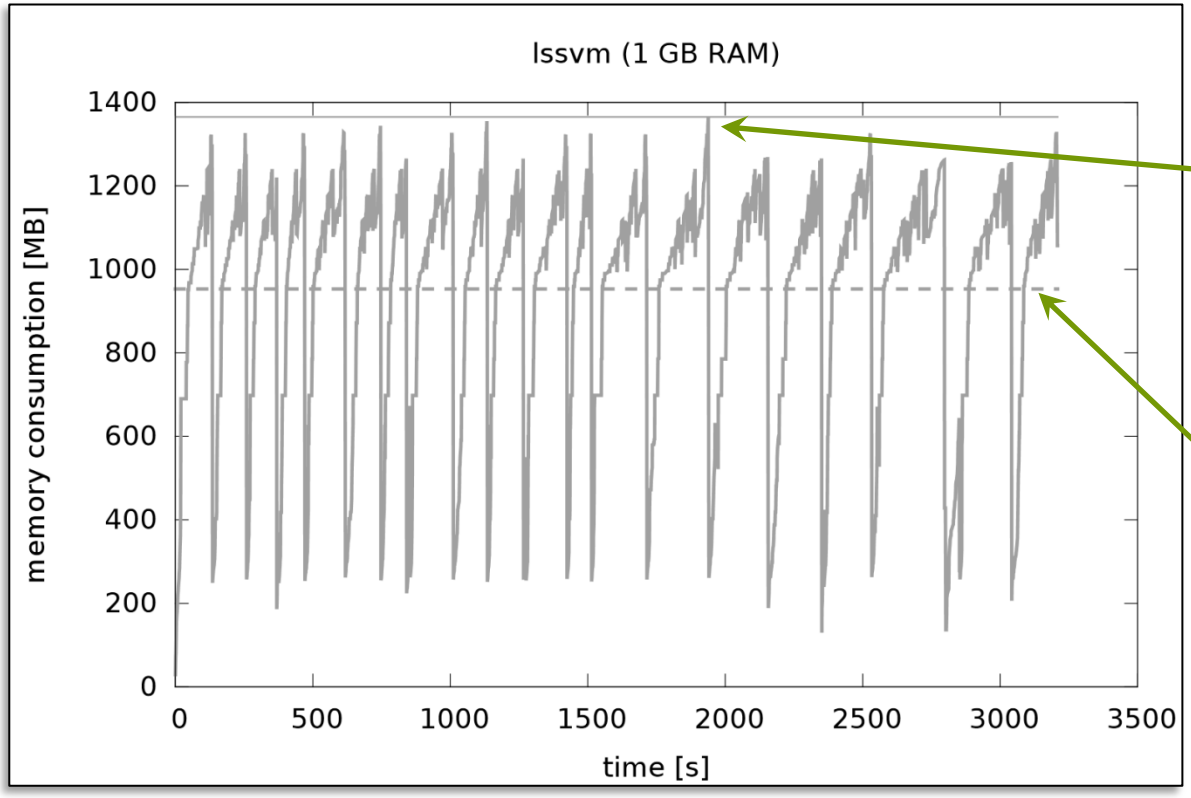
Memory Consumption Analyses for R

- *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



Helena Kotthaus
Computer Science XII

Memory-over-Time Profile

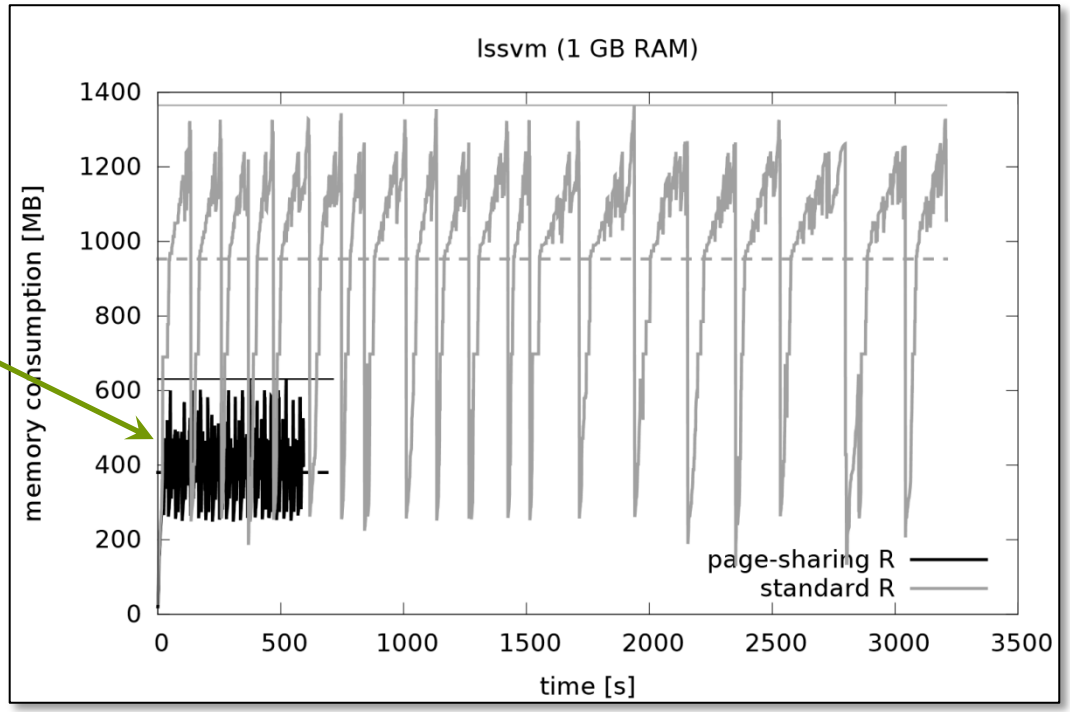


→ Indicates if your program has a *memory leak*

→ 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%*



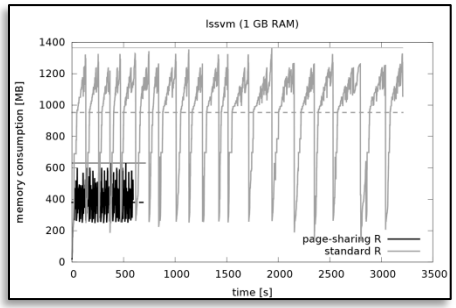
Dynamic Page Sharing Optimization for the R Language H. Kotthaus, I. Korb, M. Engel, P. Marwedel, *submitted to* Dynamic Languages Symposium

→ 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

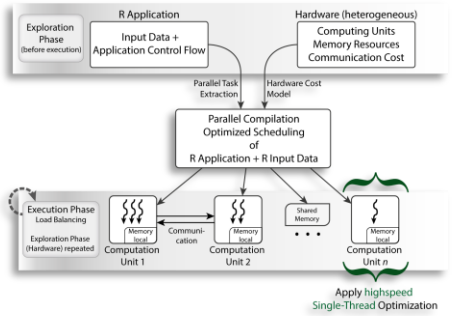
- TraceR – goal:
  → Uncover bottlenecks of R Programs and support the development of R interpreters



- Download & Install:
  → *https://github.com/allr/traceR*

- Benchmarks:
  → *https://github.com/allr/benchR*

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



Helena Kotthaus
Computer Science XII