Benchmarks is a key ingredient in performance evaluations. In this chapter, several benchmarks for different code generation methods and variations of runtime structures are described. Then the effects of different optimization levels in the \texttt{rm12c} compiler are evaluated, followed by some comparisons between RML, Typol, two commercial Prolog compilers, and one commercial Pascal compiler.

## 10.1 Target Systems

The \texttt{rm12c} compiler should run on any machine supported by recent versions of Standard ML of New Jersey (SML/NJ), as explained in Section 5.3. The generated code and the runtime system is known to work on several combinations of architectures, operating systems, and compilers. The following targets were used for the benchmarks:

- DECstation 3000/300 with a 150MHz Alpha processor and 48MB memory, OSF/1 V1.3A operating system, DEC C compiler version 3.11 (flags \texttt{-std1 \ -O2 \ -Olimit 1500}) and Gnu C compiler 2.7.0 (flags \texttt{-O2}). The nicknames for these targets are \texttt{alpha-cc} and \texttt{alpha-gcc}.

- HP 9000/715/75, 75MHz PA-RISC, 64MB, HP-UX A.09.05, HP C (flags \texttt{-Aa \ -D_HPUX\_SOURCE +O2 +Onolimit +ESlit}) and GCC 2.7.0 (flags \texttt{-O2}). Nicknames \texttt{hp-cc} and \texttt{hp-gcc}.

- HP PC, 66MHz Pentium, 32MB, Solaris 2.4, Sun ProCompiler C 2.0.1 (flags \texttt{-xO2}) and GCC 2.7.0 (flags \texttt{-O2}). Nicknames \texttt{i386-cc} and \texttt{i386-gcc}.

- DECstation 5000/200, 25MHz MIPS R3000, 56MB, Ultrix V4.3, Mips C 2.10 (flags \texttt{-O2 \ -Olimit 1600}) and GCC 2.7.0 (flags \texttt{-O2}). Nicknames \texttt{mips-cc} and \texttt{mips-gcc}.

- Parsytec GigaCube PowerPlus with 128 80MHz PowerPC 601 and 2GB memory (organized as a network of 64 nodes, each with 2 PowerPCs and 32MB memory), PARIX 1.3.1, Motorola C 1.5 (flags \texttt{-O3 \ -w \ -Ac=mixed})
and GCC 2.7.0 (flags -O2 -mno-stack-check). Nicknames ppc-cc and ppc-gcc. For the benchmarks, only a single processor on a single node was used.

Sun SPARCstation 10/40, 40MHz SuperSPARC, 64MB, Solaris 2.4, Sun C 3.0.1 (flags -xO2) and GCC 2.6.3 (flags -O2). Nicknames sparc-cc and sparc-gcc. (The code is also known to work under SunOS 4.1.3 with GCC 2.6.3.)

C compiler bugs made some targets less than fully operational:

• GCC (both 2.6.3 and 2.7.0) generates incorrect code for the ‘plain’ and ‘switch’ runtime systems on the Intel x86 architecture.

• The HP C compiler sometimes generates incorrect code for the ‘pushy’ runtime system, which caused the ‘miniml’ and ‘petrol’ examples to fail.

10.2 Overview

The three RML example specifications, mf, miniml, and petrol, were used for the benchmarks. The inputs were fixed as follows:

• The mf interpreter was applied to a small functional program computing the first 60 primes using the sieve method. The representation of the test case was built into the interpreter itself, thus obviating any overhead from scanning or parsing.

• The miniml type-checker was applied to a 154 line SML program containing 38 definitions of various more-or-less polymorphic functions (combinators, Peano arithmetic, lists, linear algebra, sorting), followed by one extremely polymorphic expression adapted from [113, Example 1.1]. (The expression has more than 4000 schematic type variables.) The reported timings do not include time spent in the scanner or parser.

• The petrol compiler was applied to a 2293 line Diesel\(^1\) program, consisting of 128 procedures and functions. (These were collected from the original Diesel compiler’s test suite.)

Due to the enormous number of combinations of different code generation options, and runtime system parameters, it is assumed that both the memory allocation arena size, and the state access method, can be measured independently.

First the effects of varying the size of the memory allocation arena for every combination of target system and code generation method will be evaluated. The best choice will then be used during the second test, which is to vary the state access methods for the ‘plain’ and ‘switch’ runtime systems.

All of the RML code tested has been compiled with all optimizations enabled. In the next test, the three optimization groups in the rml2c compiler are evaluated by disabling each one separately, measuring the effect on code

\(^1\)Recall that Petrol is a proper extension of Diesel.