Optimization in code generation to reduce energy consumption

David Branco
Mestrado em Engenharia Informática

Supervisor:
Pedro Rangel Henriques

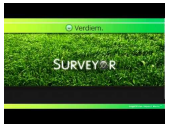
University of Minho
November 2018

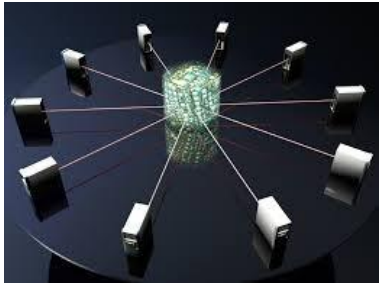
Problem and Motivation

The rapid development & high demands in computer technology, the fast growth of the Internet & massification widespread of advanced wireless and mobile devices, produce:

- Increasing energy costs;
- Increasing cooling requirements;
- Increasing equipment power density;
- Restrictions on energy supply and access;
- Growing awareness of IT impact on the environment.

Measures and Solutions





Our Project

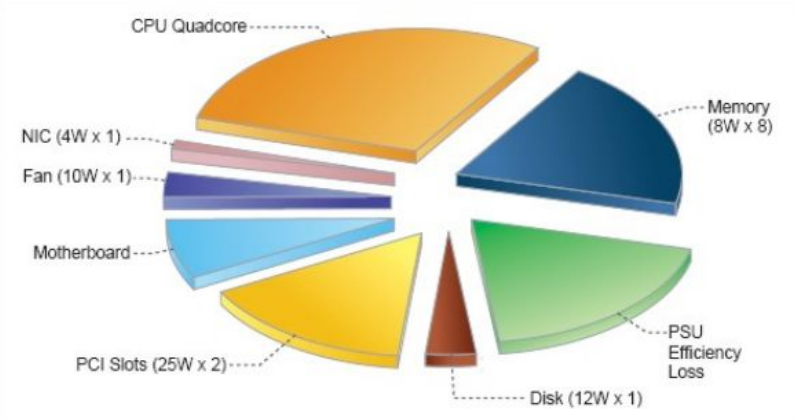
Main objectives:

1. Study the Green Computing paradigm;
2. Investigate IT elements from the energy efficiency perspective;
    2.1. Hardware;
    2.2. Software;
3. Develop an experimental study related to the triplet hardware-software-energy;
4. Produce suitable elements for other green oriented research.

Why Microprocessors ?

- They are everywhere !

- High energy consumption



Why Compilers ?

- Importance, robustness and maturity;
- Optimal properties of generated code:
  - Correctness;
  - Small Size;
  - High Speed;
  - Low Energy Consumption;
- Potential to achieve significant improvements through optimizations:
  - ✔ Execution Time;
  - ✔ Code Size;
  - (?) Energy Consumption;

Study 1 | Objectives

Impact of GCC optimization levels in energy consumption during program execution

Main Objectives:

- Impact of optimized code on CPU energy consumption;
- Ascertain the strategy adopted by GCC regarding energy efficiency.

Study 1 | Experimental Elements

Main Elements:

- Execution time and energy consumption;
- *3* hardware components: CPU, RAM e GPU;
- *12* benchmarks in *4* programming languages: C, C++, Objective-C and Go;
- GCC optimization levels: O0, O1, O2, O3, Ofast and Os.

Measurement Elements:

- Target Machine with Intel CPU (Haswell Family);
- Measurement Tool using RAPL.

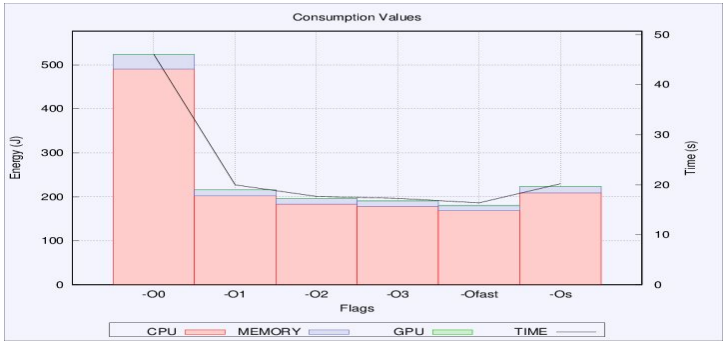
Study 1 | Methodology

Measurement Process:

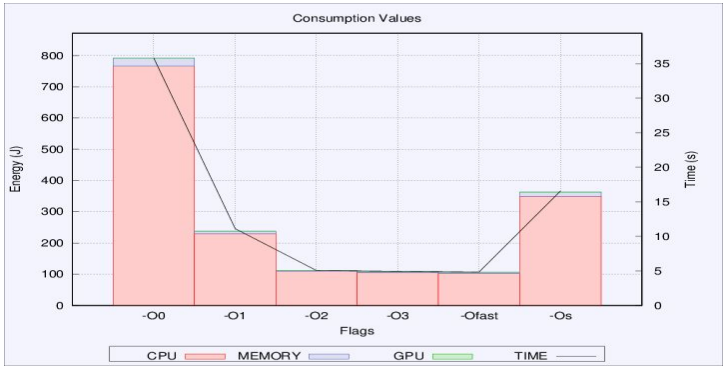
1. Select a program;
2. Select an optimization level;
3. Apply the measurement tool *100* times for the selected components;
4. Process the output generated:
   4.1. Exclude the *20* extreme cases;
   4.2. Calculate the intended average values;
5. Repeat the process for all the remaining programs and compilation profiles;
6. Output process to charts, tables and HTML pages.

Study 1 | Results

Oggenc (C)

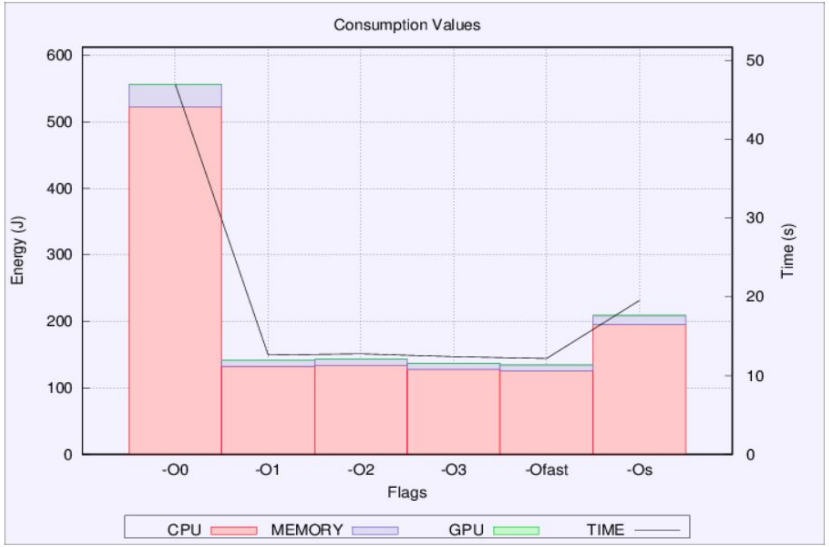


PGo (Go)



Pbrt (C++)

Flags	-O0	-O1	-O2	-O3	-Ofast	-Os
Time (s)	47.008	12.614	12.761	12.413	12.175	19.547
GPU (J)	0.028	0.006	0.005	0.096	0.220	0.009
Memory (J)	34.017	9.143	9.250	9.010	8.852	14.158
CPU (J)	522.205	132.318	133.565	127.830	125.526	195.006



Study 1 | Conclusions

Some of Study Conclusions:

- Optimization levels allow achieve considerable improvements;
- Correlation between execution time and energy consumption;
- Ofast > O2, O3 > O1, Os;
- Optimization results transverse to the different languages;
- It is not possible to conclude with certainty GCC's strategies on the matter.

Study 2 | Objectives

Impact of compilation by IDEs in energy consumption during program execution

Main Objectives:

- Deepen research on the compilation parameters;
- Study and evaluate software development tools;
- Perform measurements on more demanding benchmarks.

Study 2 | Tools, Profiles and Parameters

Software Development Tools:

- Provide 2 profiles: Debug and Release;
- Little diversity and quantity:
  - *51* profiles: *29* distinct (43% repeated);
  - *144* parameters: *28* distinct (81% repeated).

Compilation Parameters:

- Configuration of the compilation process (31%);
- Management of diagnostic messages (26%);
- Produce debugging/profiling information (17%);
- Optimization of generated code (26%).

Study 2 | Experimental Elements

Main Elements:

- *12* C benchmarks;
- *18* Software Development Tools (*15* IDEs e *3* BATs);
- *51* compilation profiles consisting of *144* compilation parameters;
- Execution time, CPU and RAM energy consumption, Energy/Time ratio.

Measurement Elements:

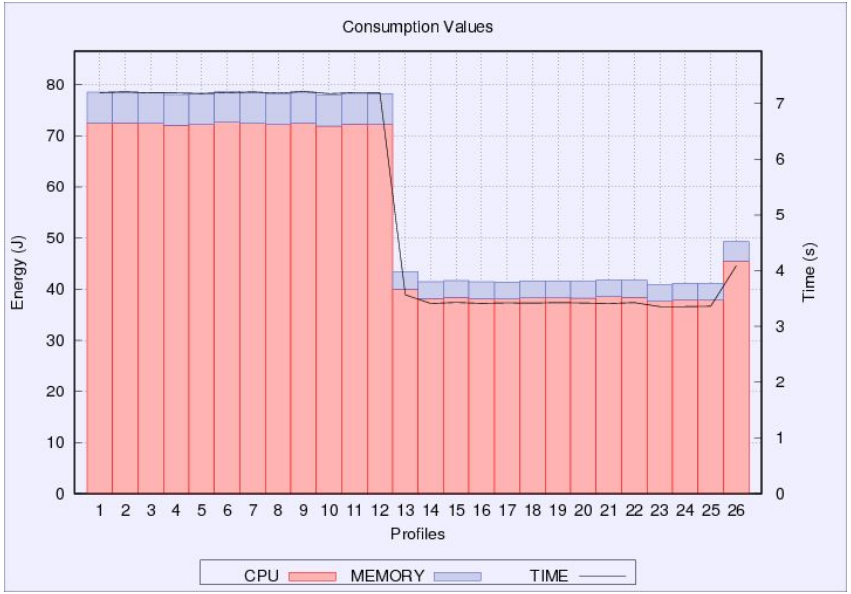
- Target Machine with Intel CPU (Haswell Family);
- Measurement Tool using RAPL.

Study 2 | Methodology

Measurement Process:

1. Select a program;
2. Select a compilation profile;
3. Apply the measurement tool *50* times for the selected components;
4. Process the output generated:
   4.1. Exclude the *20* extreme cases;
   4.2. Calculate the intended average values;
5. Repeat the process for all the remaining programs and compilation profiles;
6. Output process to charts, tables, rankings and HTML pages;

Study 2 | Profiles and Parameters Results

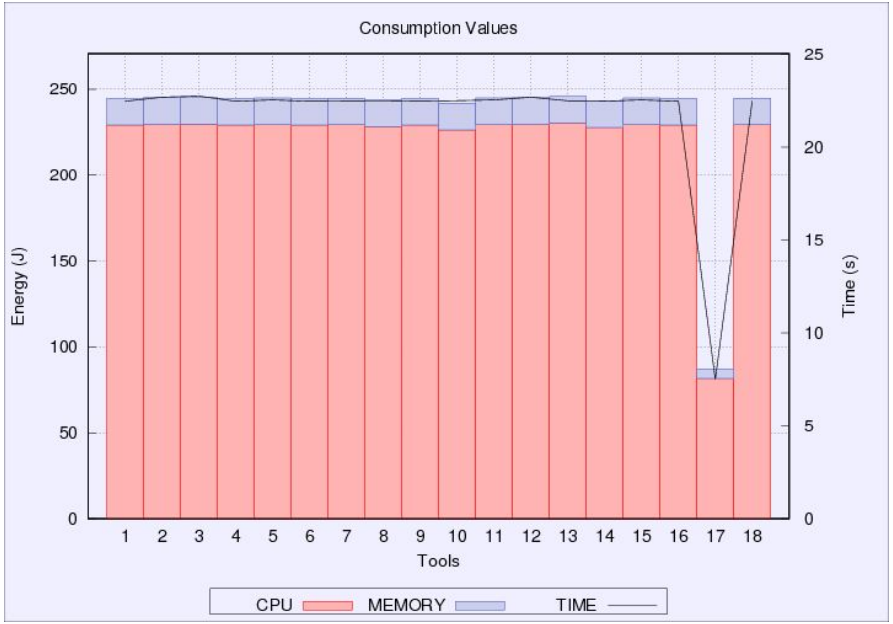


Total Profiles - Binary-trees

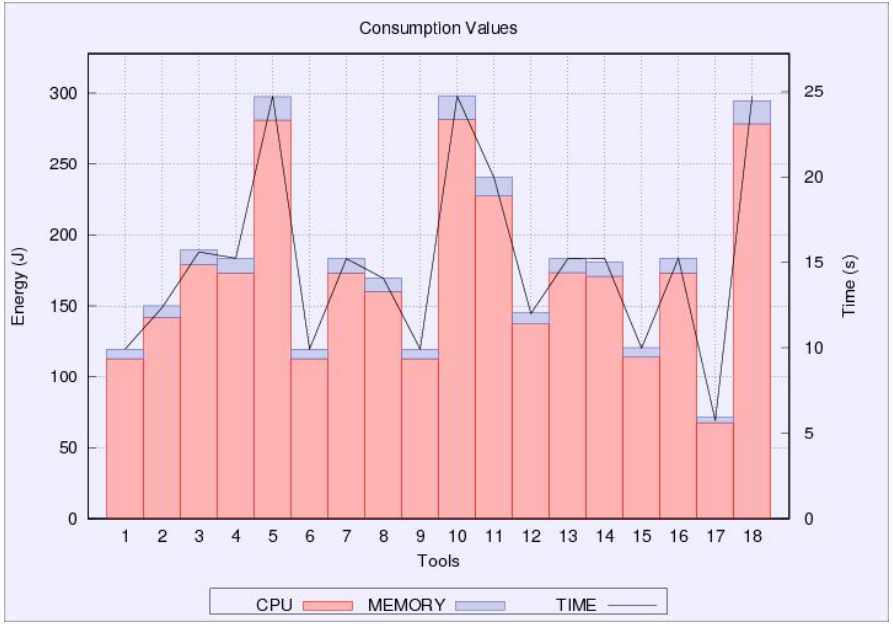
Optimization Level	Time (s)	Energy (J)	CPU (J)	Memory (J)	Energy/Time (J/s)
O0	16.862	181.611	169.872	11.739	10.582
O1	8.689 48.5%	85.511 52.9%	79.195 53.4	6.316 46.2%	10.212 3.5%
O2	8.415 50.1%	82.182 54.7%	76.063 55.2	6.119 47.9%	10.032 5.2%
O3	7.644 54.7%	74.058 59.2%	68.485 59.7	5.573 52.5%	9.872 6.7%
Os	9.408 44.2%	93.879 48.3%	87.061 48.7	6.818 41.9%	10.315 2.5%

Comparison between optimization levels

Study 2 | Tools Results



Default Profiles - K-nucleotide



Total Tools - N-body

Study 2 | Tools Results

Tool ID	Tool Name	Execution Time (s)	Total Energy (J)	CPU Energy (J)	Memory Energy (J)	Energy/Time (J/s)
1	CMake	3	3	3	3	3
2	qmake	8	4	4	6	4
3	Qbs	11	9	9	11	7
4	NetBeans IDE	10	11	9	10	7
5	Code::Blocks	15	15	14	15	11
6	CLion	3	3	3	3	3
7	CodeLite	6	7	7	7	8
8	Eclipse CDT	5	6	6	5	6
9	KDevelop	3	3	3	3	3
10	Geany	14	13	12	14	12
11	Anjuta DevStudio	12	12	11	12	9
12	Qt Creator	7	5	5	8	5
13	DialogBlocks	9	10	10	9	9
14	ZinjaI	4	8	8	4	10
15	GPS	2	2	2	2	2
16	Oracle Developer Studio	10	11	9	10	7
17	Sphere Engine	1	1	1	1	1
18	AWS Cloud9	13	14	13	13	13

Tools ranked with 1 decimal point

Study 2 | Conclusions

- Some tools have good solutions for developers;

- Measured Strands:
  - Great impact of the compiler (*86%* for the best case);
  - CPU is responsible for *90%* of energy consumption.
  - Correlations Observed:
    - Execution time and total energy consumption;
    - Total and CPU energy consumption;
    - Execution time and memory energy consumption.

- Maximum energy efficiency achieved:
  - CPU(53%) > Total energy(52%) > Execution time(47%) > RAM(46%).

Our Contribution

Some of project contributions:

1. Green oriented research of IT components;
2. Definition of experimental studies and methodologies;
3. Relevant results and conclusions;
4. Measurement Tool;
5. Green oriented workbench.

All this material is available at:

http://www4.di.uminho.pt/~gepl/OCGREC/

Optimization in code generation to reduce energy consumption

David Branco
Mestrado em Engenharia Informática

Supervisor:
Pedro Rangel Henriques

University of Minho
November 2018