Optimization of Mono on x86/Power

Progress Update

INTRODUCTION

•Our project is based on Mono, a framework to build cross platform applications.

• Mono is the open source implementation of the Microsoft's .NET framework.

• The compilation engine of Mono comprises of the Just in time compiler, partial and full Ahead of time compilation

• CIL- common intermediate language is the instruction set to which programming languages are converted, which is then executed by the compilation engine.

• Today many compilers that target the CIL are known to generate fairly straightforward CIL code leaving most of the work to the JIT engine.

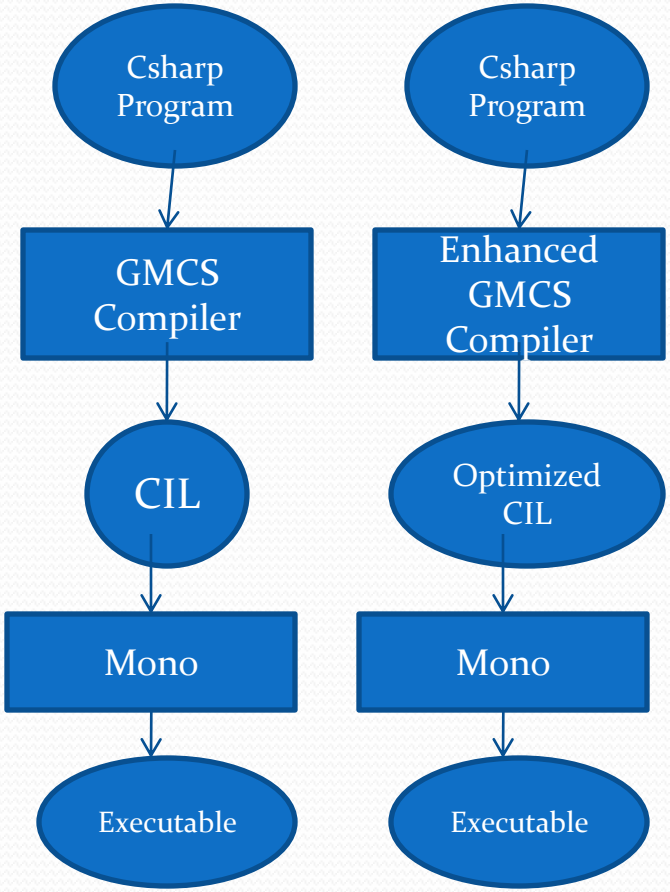
Problem Statement

- Building a code optimizer, to be integrated in Mono, that converts the CIL stream created by the GMCS compiler, into an optimized CIL stream that is in a form compatible with the JIT compiler. The code optimizer implements loop unrolling and loop fusion.

CECIL

- CECIL is an API, which allows us to :
  - read the CIL
  - browse through it
  - modify the CIL
  - save back the assembly to the disk.

Enhanced GMCS



Before Integration

After Integration

Factors on which loop unrolling depends

- Number of iterations
- Size of the unrolled loop body

Benefits of Loop Unrolling

- Parallel execution of instructions within loop body
- Reduction in loop overhead

Process of Loop Unrolling

- We read the instruction which pushes the amount to be incremented and replaced this with an increment of 2.
- The body within the loop is found between the branch instruction mentioned above and the check condition
- We stored the position of both these instructions and insert before the check condition the new loop body.

Performance analysis of loop unrolling

Array length	Performance without loop unrolling(ns)=ti	Performance with loop unrolling(ns)=t2	Performance gain of loop unrolling (t1/t2)
12	21.9	15.95	1.373
10000	11753	9733	1.207
10000000	2571063	2627573	0.9784

Table 1. Performance without and with loop unrolling with variable iterations

Demo: Performance gain shown by the integrated framework

Δx: overhead of loop unrolling in code optimizer
Δg: performance improvement due to loop unrolling

nΔg- Δx : Observed improvement
Where n is the number of times the loop is called

Performance analysis of the integration of loop unrolling into mono

Number of times loop is called	Performance script unmodified(ns)=t1	Performance of Enhanced GMCS (ns)=t2	performance gain(t1/t2)
10,000,000	672967157	879724338	0.764
100,000,000	4076261625	4133325161	0.987
1,000,000,000	40110794125	36399792991	1.101

Table 2 Comparing performance of before and after integration of loop unrolling

Benefits of Loop Fusion

- When two loops use the same data, it is beneficial to reduce the amount of data accessed between the loops.
- Cache lines fetched into the cache by the first loop may then still be in the cache when the second loop is run, so that it doesn't have to fetch them.
- Reduction in loop overhead

Process of Loop Fusion

- We check if the loops to be fused, have the same amount to be incremented and the same loop bound.

- If they do, we insert the loop body of the second loop into the loop body of the first, and remove the body of the second loop.

Demo: Performance gain shown by the integrated framework

Δx: overhead of loop fusion in code optimizer
Δg: performance improvement due to loop fusion

nΔg- Δx : Observed improvement
Where n is the number of times the loop is called

Performance analysis of the integration of loop fusion into mono

Number of times loop is called	Performance script unmodified(ns)=t1	Performance of Enhanced GMCS (ns)=t2	performance gain(t1/t2)
10,000,000	928526112	1096921133	0.846
100,000,000	6689564996	5674842597	1.17
1,000,000,000	63264124735	58812566270	1.07

Table 3. Performance before and after loop fusion

Further Work

- Our work has brought in a middle box into mono, which is the code optimizer. So, further optimizations can be added into this box.
- Loop detection may be done more accurately
- More kind of loops may be added to be optimized
- More loop optimizations may be included
- This entire frame comprising of ‘gmcs-code optimizer-mono’ can be made to run on the power PC architecture or any other architecture., as loop optimizations are architecture independent optimizations

Thank You