Compiler Optimizations for Performance

Wei Li

Example I: Computer Games

Pocket Quake* II - Compiler Benchmark The demo compares two compiled versions of the same benchmark. The better optimized version provides about 2X performance boost on Pocket Quake II.



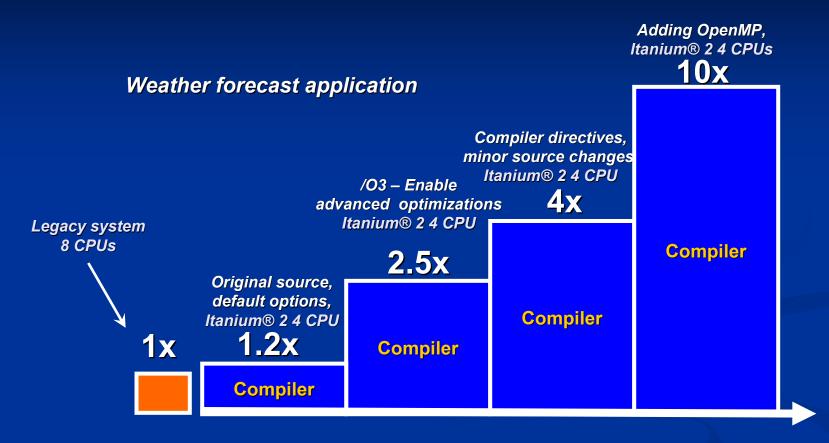


Click object in Powerpoint presentation mode to start film

Less optimizations

More optimizations

Example II: Weather Forecast



Optimizations may improve performance significantly.

Compiler Features (example)

- O2: scalar optimizations and basic scheduling
- O3: optimizations for technical computing applications (loopy codes)
- O1: optimizations for server applications (straight-line and branchy codes with flat profile)

Compiler Features (example)

- PGO (profile-guided optimizations): using profile to guide optimizations.
- IPO (inter-procedural optimizations): multi-file inlining, interprocedural optimizations.
- Parallel: automatic parallelization
- OpenMP: exploiting thread-Level parallelism
- Optimization report: compiler optimizations performed or not done

Compiler Organization (example)



Interprocedural analysis and optimizations: inlining, constant prop, whole program detect, mod/ref, points-to

Loop optimizations: data deps, prefetch, vectorizer, unroll/interchange/fusion/dist, auto-parallel/OpenMP

Global scalar optimizations: partial redundancy elim, dead store elim, strength reduction, dead code elim

Code generation: predication, software pipelining, global scheduling, register allocation, code emit

Disambiguation:

types, array, pointer, structure, directives

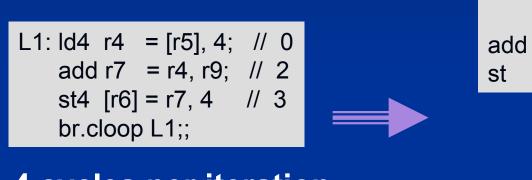
Example: Software Pipelining

```
L1: ld4 r4 = [r5], 4; // 0
add r7 = r4, r9; // 2
st4 [r6] = r7, 4 // 3
br.cloop L1;;
```

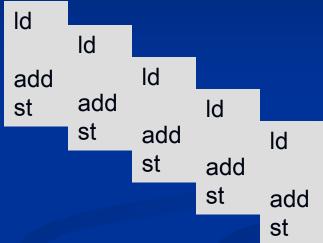
4 cycles per iteration

Example: Software Pipelining

• Exploit parallelism across loop iterations.



4 cycles per iteration



Example: Software Pipelining

• Exploit parallelism across loop iterations.

```
Id
                                                  Id
L1: Id4 r4 = [r5], 4; // 0
                                                        Id
                                            add
    add r7 = r4, r9; // 2
                                                  add
                                            st
                                                               Id
    st4 [r6] = r7, 4 // 3
                                                  st
                                                        add
                                                                     Id
    br.cloop L1;;
                                                        st
                                                              add
                                                              st
                                                                    add
4 cycles per iteration
                                                                    st
```

```
L1:
(p16) ld4 r32 = [r5], 4  // cycle 0
(p18) add r35 = r34, r9  // cycle 0
(p19) st4 [r6] = r36, 4  // cycle 0
br.ctop L1;;  // cycle 0
```

1 cycle per iteration (with architecture support)

Course Emphasis

- Compiler foundation
 - Theoretical frameworks
 - Algorithms
- Experimentation
 - Hands-on experience
 - Non-goal: how to build a complete optimizing compiler
- Exposure to real world impact
 - How they worked in practice