HITACHI
Inspire the Next

Interprocedural Analysis and Parallelization

Makoto Satoh, Yuichiro Aoki, Motoyasu Takabatake [1) 2)] , Takayoshi Iitsuka[2)], and Sumio Kikuchi[3)]

1) Advanced Parallelizing Compiler Project

2) Systems Development Laboratory (Hitachi, Ltd.)

3) Software Development Center (Hitach, Ltd.)

HITACHI
Inspire the Next

Contents

Introduction

HITACHI
Inspire the Next

Phenomena in natural / social sciences

large complex

Simulation program

large complex

extract parallelism

- Laborious task
  - large programs
  - error-prone
- Difficult task
  - data dependence
  - variable attributes

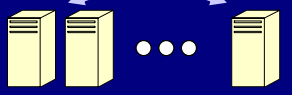
Parallelizing compiler

- Automatic
- High Performance

OpenMP program

portable scalable

Our Goal



SMPs

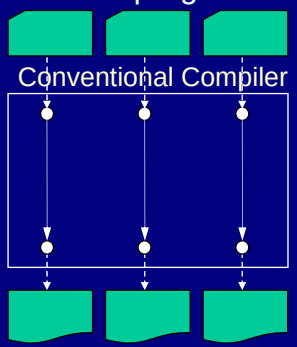
Hitachi company Z

Features of WPP* (1)

HITACHI
Inspire the Next

Conventional Compiler:
Intra- procedural Compilation

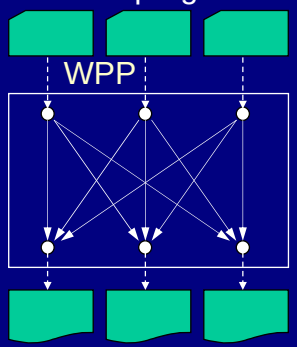
Serial program



Ineffective OpenMP program

WPP module:
Inter- procedural Compilation

Serial program



High-Performance OpenMP program

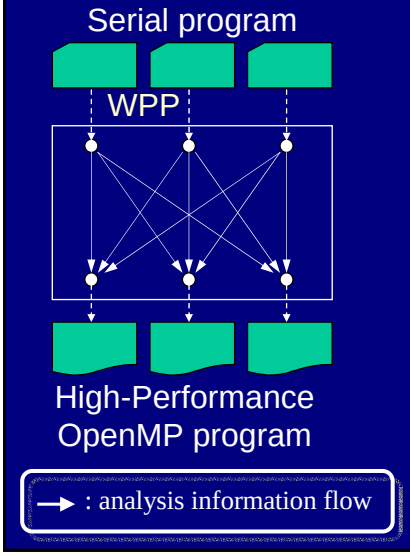
→ : analysis information flow

***: Whole Program Parallelizer**

Features of WPP (2)

HITACHI
Inspire the Next

WPP module:
Inter-procedural Compilation

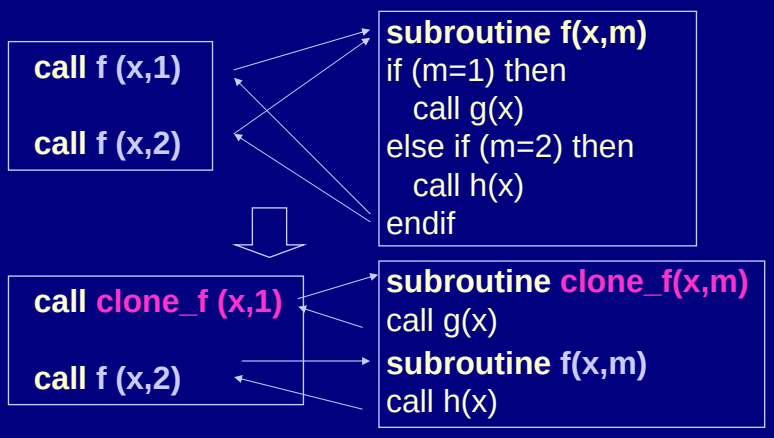


- **Interprocedural Optimization**
  - * Aggressive constant propagation
- **Interprocedural Analysis**
  - * Array region analysis
  - * COMMON variable analysis
- **Interprocedural Parallelization**
  - * Scalar/Array privatization
  - * Reduction
- **Cooperative Parallelization**
  - *** OpenMP + (Auto) Interproc.**
  - *** Intra- + Inter-procedural**
- **OpenMP program generation**

HITACHI
Inspire the Next

Aggressive Constant Propagation

**The WPP makes clone procedures,
propagates constants beyond procedure boundaries,
and evaluates expressions at compile time.**

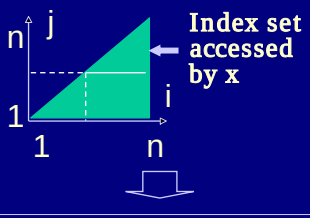


HITACHI
Inspire the Next

Array Region Analysis

The WPP uses a system of linear inequalities that enables the analysis of complex access patterns and complex argument-to-parameter transformation.

```
do j=1, n
   do i = j, n
      x(i, j) = …
```



{ (i, j) | 1≦i≦n, 1≦j≦n, j≦i }

Access pattern info for x

subroutine P

argument

A(4, 12)

Transf. of index

subroutine Q

parameter

A(4, 4, 3)

HITACHI
Inspire the Next

COMMON Variable Analysis

The WPP analyzes reference and alias information of COMMON variables; it can change the variable attributes at every subroutine (furthermore, at every loop).

subroutine P

COMMON variable

A(1..8) B(1..8)

Attribute of B → local, private, …

subroutine Q

COMMON variable

X(1..8)

subroutine P

COMMON variable

A(1..8) B(1..8)

Difficult to change attribute

subroutine Q

COMMON variable

X(1..16)

HITACHI
Inspire the Next

Interprocedural Parallelization

The WPP can parallelize loops including procedure calls changing variable attributes to private or reduction if necessary.

```
do i=1, n
   ...
  call f(x,y)
   ...
end do
```

```
subroutine f(x, y)
        …
  x = x + 2.0
        …
```

```
!$OMP PARALLEL
!$OMP DO REDUCTION(+: X)
      do i=1, n
         ...
        call f(x, y)
         ...
      enddo
!$OMP END DO NOWAIT
!$OMP END PARALLEL
```

HITACHI
Inspire the Next

Cooperative Parallelization (1)

The WPP can parallelize both of the loops to which OpenMP is specified and the loops not specified in a program and enclose the contiguous parallel loops in a parallel region.

```
      do i=1, n
        call f(x, y)
      enddo

!$OMP PARALLEL
!$OMP DO
      do j=1, n
         ...
      enddo
!$OMP END DO
!$OMP END PARALLEL
```

```
!$OMP PARALLEL
!$OMP DO
      do i=1, n
        call f(x, y)
      enddo
!$OMP END DO NOWAIT
!$OMP DO
      do j=1, n
         ...
      enddo
!$OMP END DO
!$OMP END PARALLEL
```

HITACHI
Inspire the Next

Cooperative Parallelization (2)

The WPP can apply inter-procedural parallelization to the loops with procedure calls and apply strong intra-procedural parallelization to the loops without them.

```
k=1
do i=1, n
  do j=k, n
    h(i, j) = …
  enddo
  k = i
enddo

do i=1, n
  call f(x, y)
enddo
```

k = 1,1,2,3, ...

1st loop is peeled and parallelized.

```
!$OMP PARALLEL
!$OMP DO
      do i=2, n
        do j=i-1, n
          h(i, j) = …
        enddo
      enddo
!$OMP END DO NOWAIT
!$OMP DO
      do i=1, n
        call f(x, y)
      enddo
!$OMP END DO
!$OMP END PARALLEL
```

HITACHI
Inspire the Next

Evaluation Environment

Hitachi SR8000

CPU: Proprietary, 375MHz
Structure: SMPs (16PEs =8PEs/node x 2 nodes)
L1: 128KB
Compiler: OFORT90 V01-04-/B

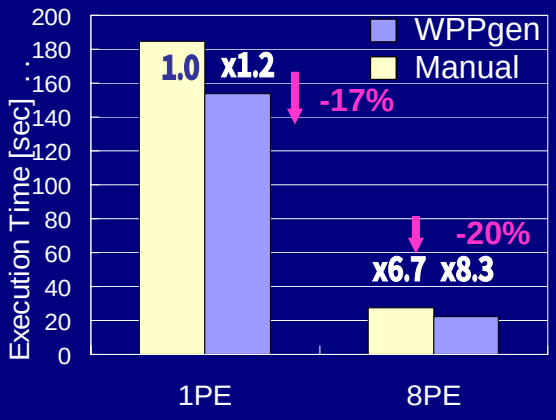
SGI® Origin® 2000

CPU: R10000®, 195MHz
Structure: DSM (32 PEs= 2PEs/node x 16 nodes)
L1: 32KB / 32KB
L2: 4MB/PE
Compiler: MIPSpro™ Fortran90 Version 7.30

Evaluation (1)

HITACHI
Inspire the Next

SPEC® CFP95 / turb3d on SR8000



* Aggr const prop
* Array region ana
- Interproc Para

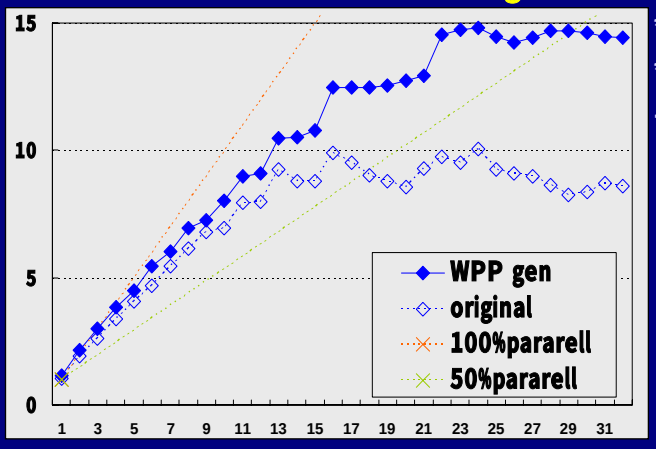
WPPgen: OpenMP program generated by WPP

Manual: OpenMP program obtained by inserting the same OpenMP directives as "WPPgen" to the original serial program.

Evaluation (2)

HITACHI
Inspire the Next

SPEC® CFP95 / turb3d on Origin® 2000



* Aggr const prop
* Array region ana
- Interproc Para

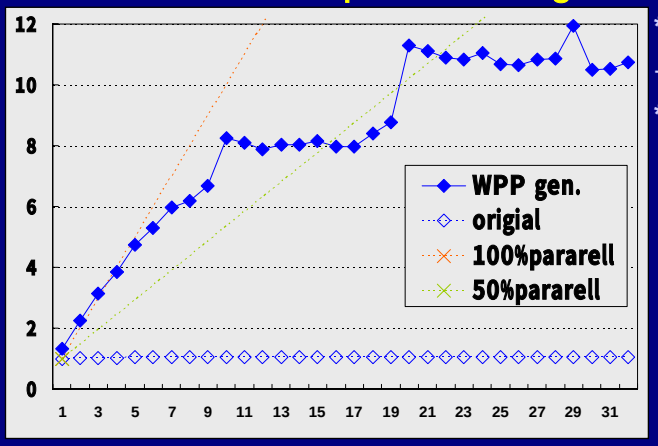
WPP gen: OpenMP program generated by the WPP

original: MIPSPro™ Fortran V.7.30

Evaluation (3)

HITACHI
Inspire the Next

SPEC® CFP2000 / wupwise on Origin® 2000



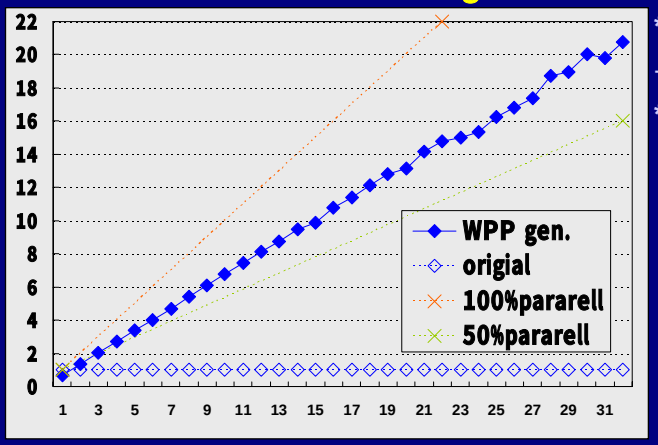
* Aggr const prop
- Interproc Para
* Private

- **WPP gen:** OpenMP program generated by the WPP
- **original:** MIPSPro™ Fortran V.7.30

Evaluation (4)

HITACHI
Inspire the Next

NPB2.3 serial / EP on Origin® 2000



* Common var ana
- Interproc Para
* Reduction

- **WPP gen:** OpenMP program generated by the WPP
- **original:** MIPSPro™ Fortran V.7.30

HITACHI
Inspire the Next

Conclusion

We have developed the WPP, Interprocedural parallelizing module in APC compiler, which

- **optimizes programs interprocedurally.**
- **parallelizes the loops including procedure calls.**
- **generates high-performance OpenMP programs.**

Evaluation shows its effectiveness in scientific benchmarks.