Data Dependency Analysis

Ruud van der Pas

SiliconGraphics

European High Performance Computing Team

Intro

- We will now try to build up *understanding* for what a compiler has to determine when deciding on parallelization
- It is important to note that when loosely talk about parallelization, it is actually meant

  "Is the program construct *as presented here* parallelizable ?"

- This is an important distinction to make, because sometimes a rewrite can transform non–parallizable constructs into a parallelizable form, as we will see ....

Data Dependency Analysis

A question

Are the following loops parallelizable?

```
DO i = 2, N
   A(i) = B(i-1)
END DO
```

```
DO i = 2, N
   A(i) = A(i-1)
END DO
```


Data Dependency Analysis

An answer

Are the following loops parallelizable?

```
DO i = 2, N
   A(i) = B(i-1)
END DO
```

YES

```
DO i = 2, N
   A(i) = A(i-1)
END DO
```

NO

WHY?

Data Dependency Analysis

```
DO i = 2, N
   A(i) = B(i-1)
END DO
```

YES

	T=1
CPU 1	A(2)=B(1)
CPU 2	A(3)=B(2)
CPU 3	A(4)=B(3)

Data Dependency Analysis

```
DO i = 2, N
   A(i) = B(i-1)
END DO
```

YES

	T=1	T=2
CPU 1	A(2)=B(1)	A(5)=B(4)
CPU 2	A(3)=B(2)	A(6)=B(5)
CPU 3	A(4)=B(3)	A(7)=B(6)

Data Dependency Analysis

```
DO i = 2, N
   A(i) = B(i-1)
END DO
```

YES

	T=1	T=2	T=3
CPU 1	A(2)=B(1)	A(5)=B(4)	A(8)=B(7)
CPU 2	A(3)=B(2)	A(6)=B(5)	A(9)=B(8)
CPU 3	A(4)=B(3)	A(7)=B(6)	A(10)=B(9)

Data Dependency Analysis

```
DO i = 2, N
   A(i) = A(i-1)
END DO
```

NO

	T=1
CPU 1	A(2)=A(1)
CPU 2	A(3)=A(2)
CPU 3	A(4)=A(3)

Will read OLD data

(probably ....)

Data Dependency Analysis

```
DO i = 2, N
   A(i) = A(i-1)
END DO
```

NO

	T=1	T=2
CPU 1	A(2)=A(1)	A(5)=A(4)
CPU 2	A(3)=A(2)	A(6)=A(5)
CPU 3	A(4)=A(3)	A(7)=A(6)

Will read NEW data

(probably ....)

Will read OLD data

(probably ....)

?

Data Dependency Analysis

More questions

Are the following loops parallelizable?

```
DO i = 3, N, 2
   A(i) = A(i-1)
END DO
```

```
DO i = 1, N
   S = S + A(i)
END DO
```


Data Dependency Analysis

More questions

Are the following loops parallelizable?

```
DO i = 3, N, 2
   A(i) = A(i-1)
END DO
```

YES

```
DO i = 1, N
   S = S + A(i)
END DO
```

Depends

WHY?

Data Dependency Analysis

```
DO i = 3, N, 2
   A(i) = A(i-1)
END DO
```

YES

	T=1
CPU 1	A(3)=A(2)
CPU 2	A(5)=A(4)
CPU 3	A(7)=A(6)

Data Dependency Analysis

```
DO i = 3, N, 2
   A(i) = A(i-1)
END DO
```

YES

	T=1	T=2
CPU 1	A(3)=A(2)	A(9)=A(8)
CPU 2	A(5)=A(4)	A(11)=A(10)
CPU 3	A(7)=A(6)	A(13)=A(12)

Data Dependency Analysis

```
DO i = 3, N, 2
   A(i) = A(i-1)
END DO
```

YES

	T=1	T=2	T=3
CPU 1	A(3)=A(2)	A(9)=A(8)	A(15)=A(14)
CPU 2	A(5)=A(4)	A(11)=A(10)	A(17)=A(16)
CPU 3	A(7)=A(6)	A(13)=A(12)	A(19)=A(18)

Data Dependency Analysis

```
DO i = 1, N
   S = S + A(i)
END DO
```

Depends

	T=1
CPU 1	S=S+A(1)
CPU 2	S=S+A(2)
CPU 3	S=S+A(3)

Which value of S will be read?

(later on we will see one way to parallelize this loop ....)

Data Dependency Analysis

```
DO i = 1, N
   S = S + A(i)
END DO
```

Depends

	T=1	T=2	T=3
CPU 1	S=S+A(1)	S=S+A(4)	S=S+A(7)
CPU 2	S=S+A(2)	S=S+A(5)	S=S+A(8)
CPU 3	S=S+A(3)	S=S+A(6)	S=S+A(9)

- *The value of S will be undetermined and typically shall vary from run to run*
- *This bug is called a "race-condition"*

Data Dependency Analysis

What is going on here?

The examples shown fall into two categories:

1. Data being read is independent of written data

   A(i) = B(i–1) i=2,3,4,....

   A(i) = A(i–1) i=3,5,7,....

2. Data being read is dependent of written data

   A(i) = A(i–1) i=2,3,4,....

   S = S + A(i) i=1,2,3,...

Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   ......
   result = A(i)+B(i)
   ......
END DO
```


Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   A(i)    = sin(x(i))
   result = A(i)+B(i)
   C(i)    = result*C(i)
END DO
```


Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   A(i)    = sin(x(i))
   result = A(i)+B(i)
   C(i)    = result*C(i)
END DO
```

NO

Clearly "result" is a temporary variable and gets re–assigned for every iteration

Note that result should be a "local" variable (don't worry, we'll talk about that in a minute).

Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   A(i)   = sin(result)
   result = A(i)+B(i)
   C(i)   = result*C(i)
END DO
```


Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   A(i)   = sin(result)
   result = A(i)+B(i)
   C(i)   = result*C(i)
END DO
```

YES

The value of "result" is carried over from iteration to iteration.

This is the classical read/write situation, but now somewhat masked !

Data Dependency Analysis

A typical situation

Is there a data dependency in this loop?

```
DO i = 1, N
   A(i)   = sin(result)
   result = A(i)+B(i)
   C(i)   = result*C(i)
END DO
```

YES

Data Dependency Analysis

The loop could (symbolically) be rewritten:

```
DO i = 1, N
   A(i)      = sin(result(i-1))
   result(i) = A(i)+B(i)
   C(i)      = result(i)*C(i)
END DO
```

Now we'll substitute the expression for A(i)

Data Dependency Analysis

```
DO i = 1, N
   A(i)       = sin(result(i-1))
   result(i)  = A(i)+B(i)
   C(i)       = result(i)*C(i)
END DO
```

```
DO i = 1, N
   A(i)       = sin(result(i-1))
   result(i)  = sin(result(i-1))+B(i)
   C(i)       = result(i)*C(i)
END DO
```

So, this is of the type "A(i)=A(i-1)" again!

Data Dependency Analysis

One More Question (and answer)

Is the following loop parallelizable?

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

If parallelized:

probably varying answers from run to run

WHY?

Data Dependency Analysis

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

Yes, but ...

	T=1
CPU 1	A(3)=A(1)
CPU 2	A(4)=A(2)

Data Dependency Analysis

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

Yes, but ...

	T=1	T=2
CPU 1	A(3)=A(1)	A(5)=A(3)
CPU 2	A(4)=A(2)	A(6)=A(4)

Data Dependency Analysis

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

Yes, but ...

	T=1	T=2	T=3
CPU 1	A(3)=A(1)	A(5)=A(3)	A(7)=A(5)
CPU 2	A(4)=A(2)	A(6)=A(4)	A(8)=A(6)

Data Dependency Analysis

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

NO !

	T=1
CPU 1	A(3)=A(1)
CPU 2	A(4)=A(2)
CPU 3	A(5)=A(3)

A(3) is read and written at the same time

Data Dependency Analysis

What is the trick?

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

We have seen that i=3 and i=5 form a so-called read/write "pair":

write: A(3) = A(1)

read: A(5) = A(3)

Question: Can this occur in general ?

Data Dependency Analysis

What is the trick?

```
DO i = 3, N
   A(i) = A(i-2)
END DO
```

Some elementary math:

W(x):= x
R(y) := y−2

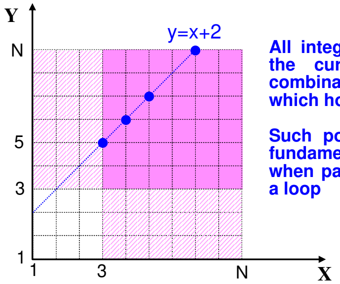
Can it happen that a pair (x,y) exists for which the following relationship holds:

W(x) = R(y) with x and y in [3,N] ⟺

x =y−2 with x and y in [3,N] ⟺

y =x+2 with x and y in [3,N]

Data Dependency Analysis



All integer points ● on the curve are index combinations (x,y) for which holds R(x)=W(y)

Such points present a fundamental problem when parallelizing such a loop

Data Dependency Analysis

What if you're a compiler?

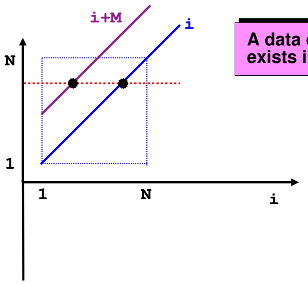
Let's examine the following loop:

```
DO i = 1, N
   A(i) = A(i+M)
END DO
```

We know that for $M \in \{-1,-2\}$ a data dependency exists

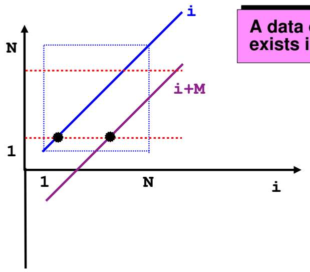
How about the more general case?

Data Dependency Analysis



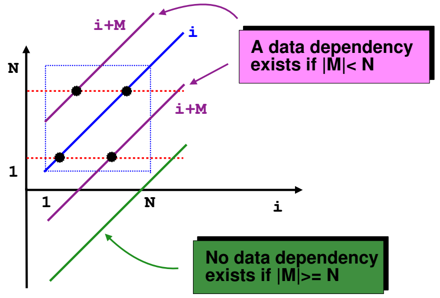
A data dependency exists in this case

Data Dependency Analysis



A data dependency exists in this case

Data Dependency Analysis



Data Dependency Analysis

What if you're a compiler?