LOOP PARALLELIZATION AND VECTORIZATION

• A loop whose dependence graph is cycle-free can be parallelized or vectorized.
  e.g.

```
DO I=1,N
   X(I)=B(I)+1
   A(I)=X(I)+1
END DO
```

```
X(1:N)=B(1:N)+1  PARALLEL DO I=1,N
A(1:N)=X(1:N)+1   X(I)=B(I)+1
                A(I)=X(I)+1
  END PARALLEL DO
```
**ALGORITHM REPLACEMENT**

- Some program patterns occur frequently in programs. They can be replaced with a parallel algorithm. e.g.

```plaintext
DO I=1,N
   A(I)=A(I-1)+B(I)
END DO

A(1:N)=REC1N(B(1:N),A(0),N)

X=A(1)
DO I=2,N
   IF(X.GT.A(I)) X=A(I)
END DO

X=MIN(A(1:N))
```
LOOP DISTRIBUTION (LOOP FISSION)

- To insulate these patterns, we can decompose loops into several loops, one for each strongly-connected component ($\pi$-block) in the dependence graph.

```
DO I=1,N
S1: A(I)-B(I)+C(I)
S2: D(I)=D(I-1)+A(I)
S3: IF(X.GT.A(I))THEN
    X=A(I)
ENDIF
END DO

DO I=1,N
A(I)=B(I)+C(I)
END DO
DO I=1,N
D(I)=D(I-1)+A(I)
END DO
DO I=1,N
IF (X.GT.A(I) THEN
    X=A(I)
END IF
END DO
```
LOOP INTERCHANGING

- The dependence information determines whether or not the loop headers can be interchanged.
- For example, the following loop headers can be interchanged

\[
\begin{align*}
d & i=1,n \\
   & \quad j=1,n \\
   & \quad a(i,j) = a(i,j-1) + a(i-1,j) \\
\end{align*}
\]

However, the headers in the following loop cannot be interchanged

![Diagram showing interchanging of loop headers](image-url)
do i=1,n
   do j=1,n
      a(i,j) = a(i,j-1) + a(i-1,j+1)
   end do
end do
**DEPENDENCE REMOVAL**

- Some cycles in the dependence graph can be eliminated by using elementary transformations.

*Scalar Expansion:*

```plaintext
DO I=1,N
  S1: A=B(I)+1
  S2: C(I)=A+D(I)
END DO

DO I=1,N
  S1: A1(I)=B(I)+1
  S2: C(I)=A1(I)+D(I)
END DO
A=A1(N)
```
Induction variable recognition

DO I=1,N
  S1: J=J+2
  S2: X(I)=X(I)+J
END DO

DO I=1,N
  S1: J1=J+2*I
  S2: X(I)=X(I)+J1
END DO

DO I=1,N
  S1: J1(I)=J+2*I
  S2: X(I)=X(I)+J1(I)
END DO