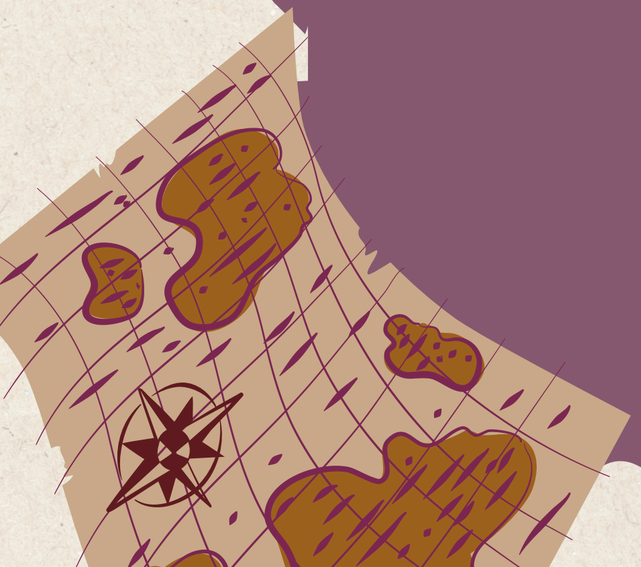
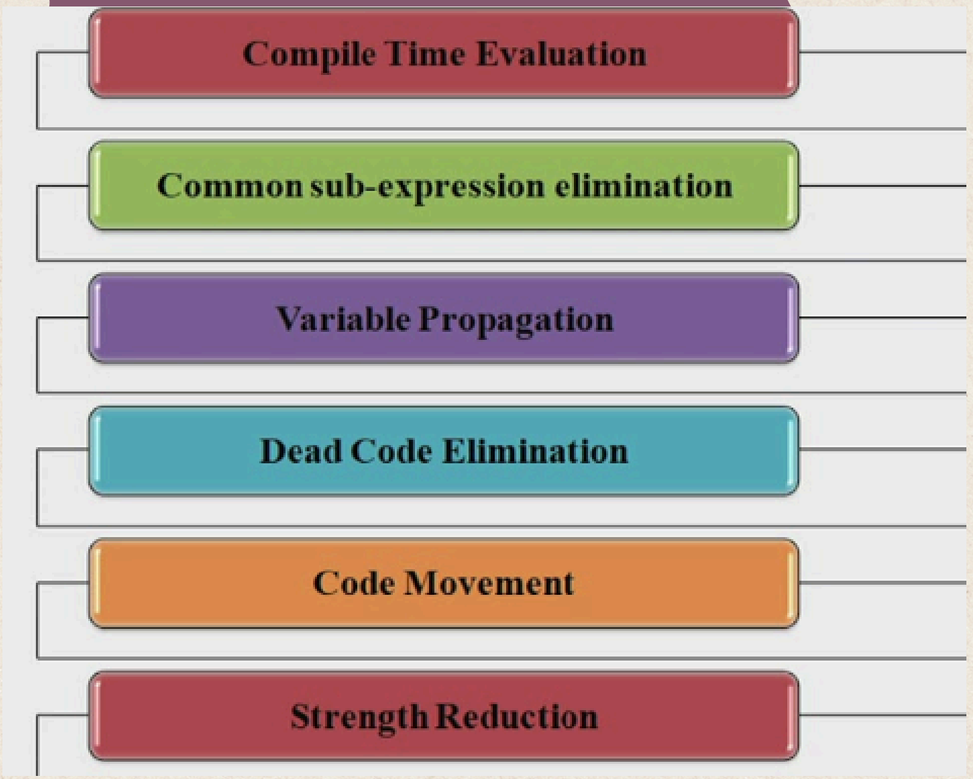
CONSTANT PROPAGATION

Simra razeen
Sreelekha



CONSTANT FOLDING IN COMPILER DESIGN



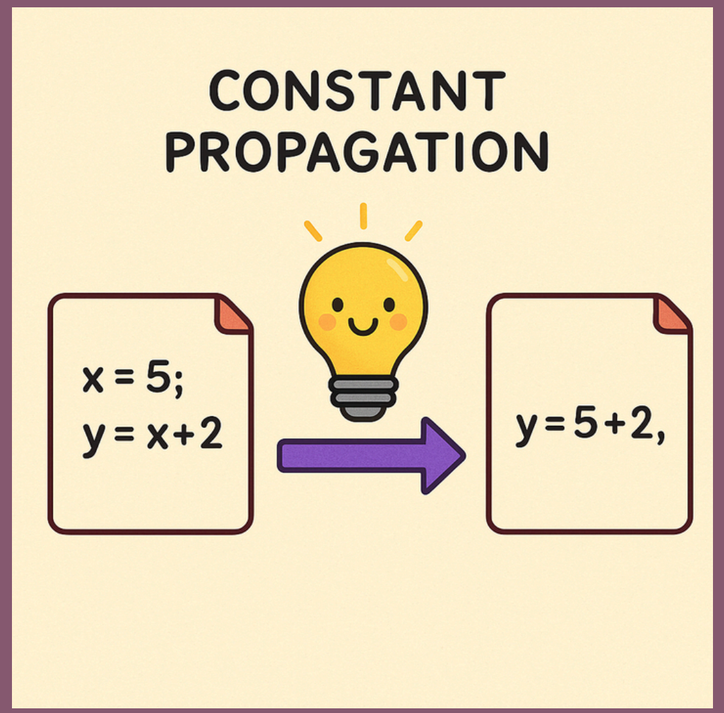
What is constant propagation ?

Constant propagation is the process of substituting the values of known constants in expressions at compile time to simplify code and improve efficiency.

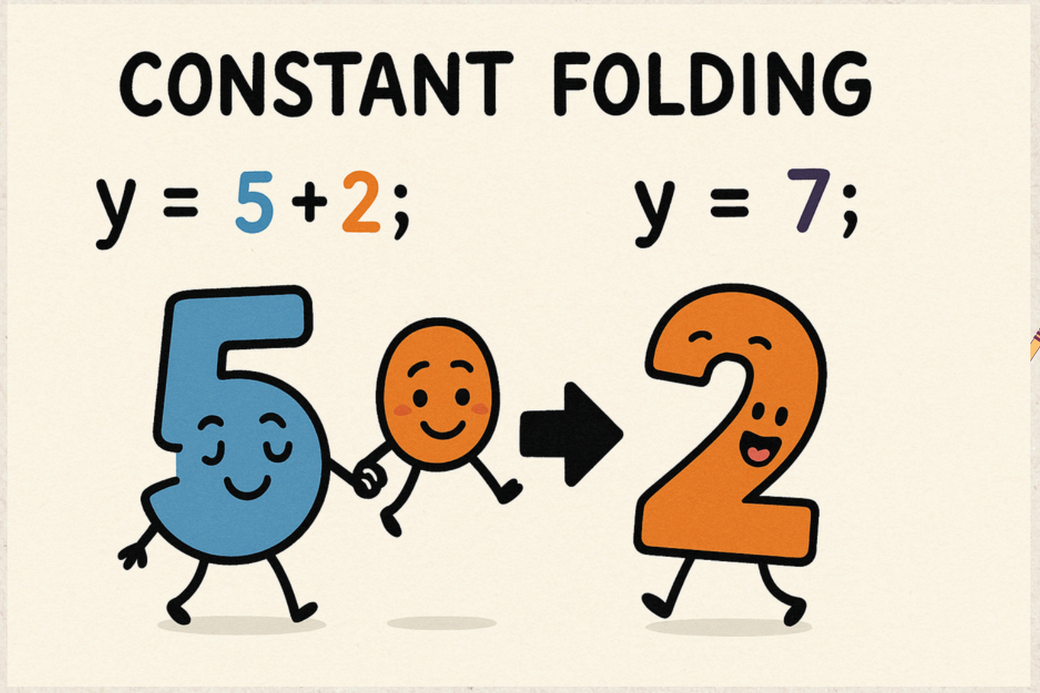
```
int a = 5;

int b = a + 3;


After Constant Propagation:
int b = 5 + 3;


After Constant Folding (optional next step):
int b = 8;
```

Constant Propagation vs. Constant Folding

Replaces variables with known constant values.

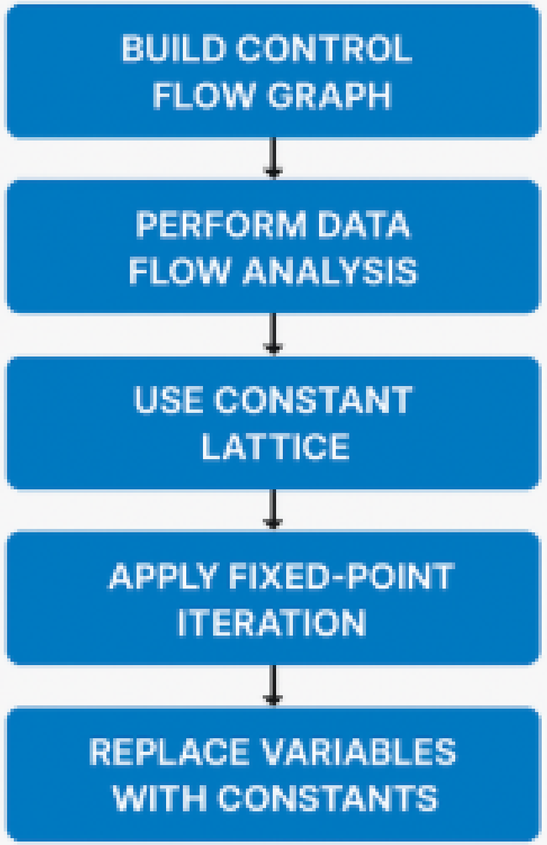


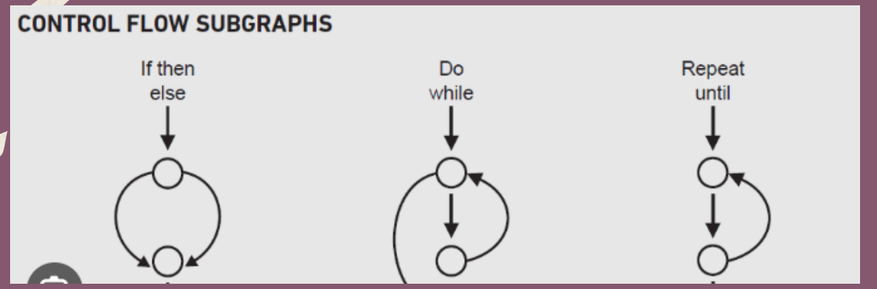
Solves constant expressions.



Both help make the code shorter and faster.

How constant propagation works?





Each variable is modeled using a **lattice** of values:

- `T` (undefined – no info yet)
- A constant value (like `5`)
- `⊥` (not a constant – value changes)

int a = 3; int b = a + 2; → becomes → int b = 5;

Step 1 :

```
w = 0;
x = x + y;
y = 0;
if( x > z)
  {
   y = x;
   x++;
  }
else
  {
   y = z;
   z++;
  }
w = x + z;
```

Source Code

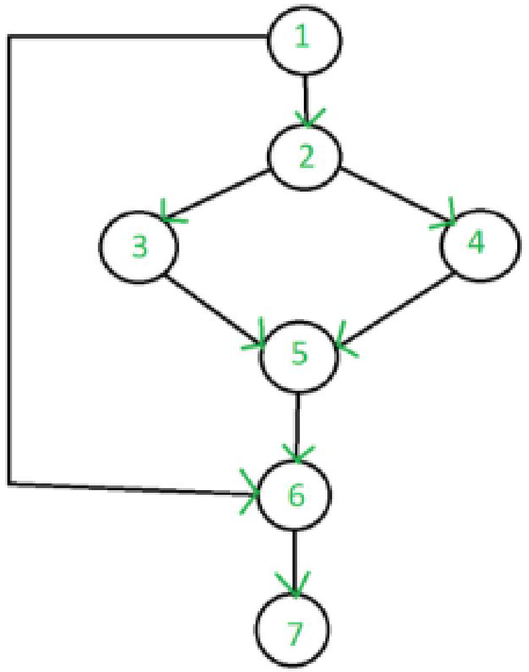
```
w = 0;
x = x + y;
y = 0;
if( x > z)
```

```
y = x;
x++;
```

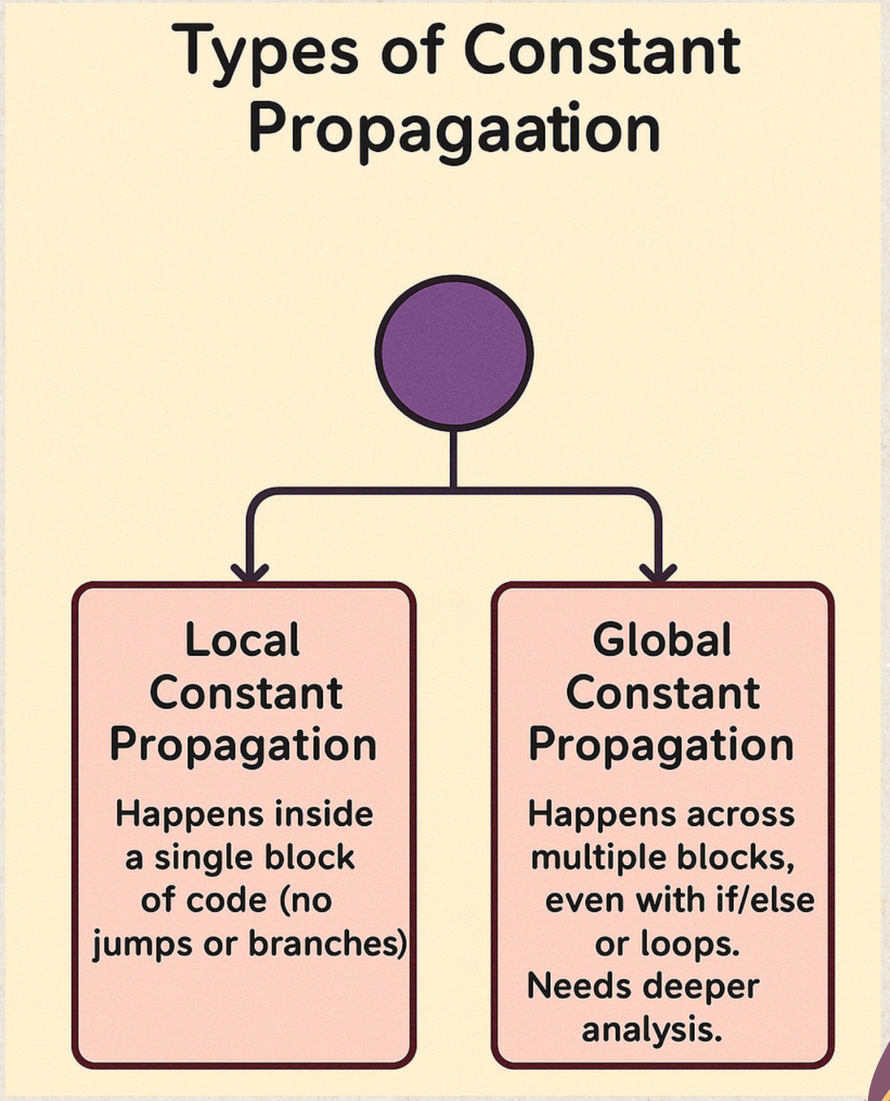
```
y = z;
z++;
```

```
w = x + z;
```

Basic Blocks



Control Flow Graph



📌 **Example:**

```
int x = 5
int y = x + 2
```

→ int y = 7

```
int x = 10;
y = x + 2
```

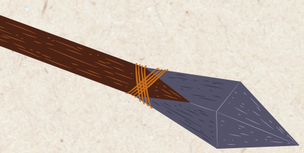
→ int y = 12

Advantages

- Improved Runtime Performance
- Enables Dead Code Elimination
- Simplifies Expressions
- Reduces Code Size
- Enables Further Optimizations
- Improves Compiler Analysis Precision

LIMITATIONS

- Works only with known values
- Can’t handle user input
- Struggles with if/else and loops
- Problems with shared variables
- Can make code bigger



Conclusion

- Compiler optimization improves program performance and efficiency by transforming code during compilation.
- Constant Propagation simplifies code by replacing variables with constant values, reducing unnecessary computations.
- While Constant Propagation optimizes constant values across the program, Constant Folding evaluates simple constant expressions at compile time.
- Understanding how Constant Propagation works helps implement more advanced optimizations like dead code elimination and loop optimization.
- Different types of Constant Propagation include basic, interprocedural, and sparse conditional, each with varying levels of complexity and efficiency.
- The advantages include reduced runtime, smaller code size, and enabling further optimizations, but there are limitations such as handling loops, function calls, and maintaining precision.
- Despite challenges, Constant Propagation remains a crucial technique in optimizing modern compilers and improving software performance.

THANK YOU