**Research Problem**

Turning mathematical graph algorithms into actual implementations that run at speed is complicated. It requires:

1. **algorithmic design** to identify the appropriate implementable algorithms
2. **tuned implementations** that consider data storage formats and available hardware features

**Target Problem**

Using triangle counting as an example, we demonstrate our approach to generating graph algorithms from their mathematical specification.

**Mathematical Specification**

\[ \Delta = \frac{1}{6} \Gamma(A^3) \]

**Proposed Solution**

Encode expert knowledge about algorithm design and optimization into an automated system (SPIRAL) to generate tuned implementations automatically.

Allow the use of GraphBLAS formulae for providing mathematical specifications.

---

**Program Specification**

**Algorithm Choice**

**Algorithm Derivation**

**Abstract Code**

**C Code**

```c
void tc(int *res, int *IJ) {
    // VMV product
}
```

---

**Research Problem**

Turning mathematical graph algorithms into actual implementations that run at speed is complicated. It requires:

1. **algorithmic design** to identify the appropriate implementable algorithms
2. **tuned implementations** that consider data storage formats and available hardware features

**Target Problem**

Using triangle counting as an example, we demonstrate our approach to generating graph algorithms from their mathematical specification.

**Mathematical Specification**

\[ \Delta = \frac{1}{6} \Gamma(A^3) \]

**Proposed Solution**

Encode expert knowledge about algorithm design and optimization into an automated system (SPIRAL) to generate tuned implementations automatically.

Allow the use of GraphBLAS formulae for providing mathematical specifications.