SITCON'21

Floyd-Warshall - GPU

- 觀察 Data dependency

$$f_k(i,j) = \min(f_{k-1}(i,j)\ ,\ f_{k-1}(i,k) + f_{k-1}(k,j))$$

- 裡面兩層 loop 沒有關聯

- 將裡面兩層迴圈平行化

```
for (int k = 0; k < n; k++) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            g[i][j] = min(g[i][j], g[i][k] + g[k][j]);
        }
    }
}
```

```
__global__ void gpu(int **g, int n, int k) {
    int id = blockIdx.x * blockDim.x+ threadIdx.x;
    int i = id / n;
    int j = id % n;
    g[i][j] = min(g[i][j], g[i][k] + g[k][j]);
}

// bs * 1024 = n^2
for (int k = 0; k < n; k++) {
    gpu<<<bs, 1024>>>(g, n, k);
}
```

Floyd-Warshall - GPU

- 觀察 Data dependency

$$f_k(i,j) = \min(f_{k-1}(i,j)\ ,\ f_{k-1}(i,k) + f_{k-1}(k,j))$$

- 裡面兩層 loop 沒有關聯
- 將裡面兩層迴圈平行化

```
for (int k = 0; k < n; k++) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            g[i][j] = min(g[i][j], g[i][k] + g[k][j]);
        }
    }
}
```

```
__global__ void gpu(int **g, int n, int k) {
    int id = blockIdx.x * blockDim.x+ threadIdx.x;
    int i = id / n;
    int j = id % n;
    g[i][j] = min(g[i][j], g[i][k] + g[k][j]);
}

// bs * 1024 = n^2
for (int k = 0; k < n; k++) {
    gpu<<<bs, 1024>>>(g, n, k);
}
```

Performance Issue

- 無連續讀取記憶體讀取 `min(g[i][j], g[i][k] + g[k][j]);`
- 無共用記憶體
- 效能瓶頸為記憶體頻寬