



AKADEMIA GÓRNICZO-HUTNICZA  
IM. STANISŁAWA STASZICA W KRAKOWIE

# **Wybrane metody automatycznej generacji kodu na procesory wielordzeniowe**

**Student realizujący: Piotr Listkiewicz  
Opiekun pracy: dr inż. Marcin Pietroń**

## Cel pracy

- **Analiza metod automatycznej detekcji kodu który można zrównoleglić**
- **Wytworzenie narzędzia umożliwiającego automatyczną generację kodu w wersji równoległej**
- **Obecnym celem jest implementacja narzędzia dla języka C , które przekształca go do postaci OpenMP**
- **Jeżeli pierwsza faza projektu poszłaby pomyślnie, to w drugiej implementacja narzędzia przekształcającego bytectomy javy do postaci równoległej przy pomocy wątków**

## Przekształcenia pętli

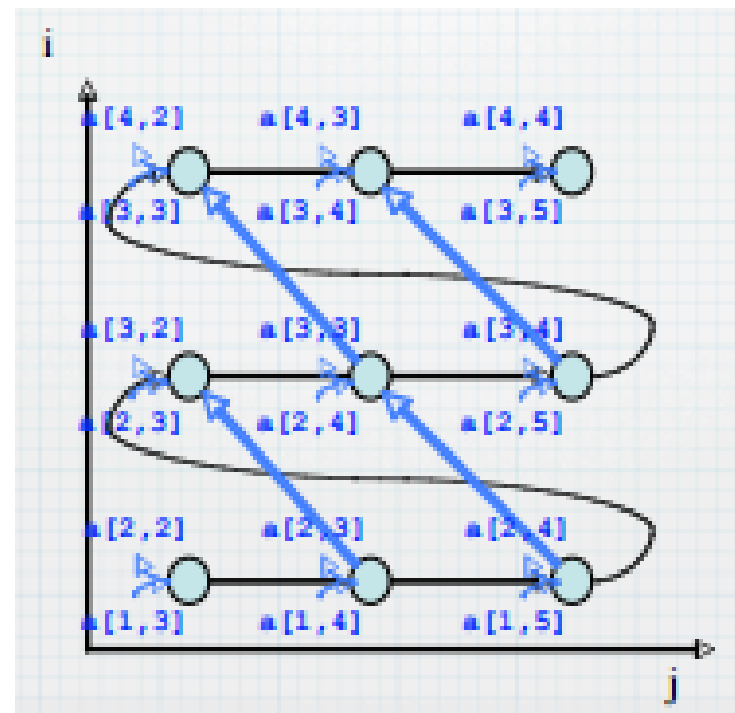
- **Loop Vectorization – przekształcenia skalarów na wektory/macierze**
- **Loop Interchanging – podmiana kolejności pętli**
- **Loop concurrentization – przypisanie iteracji do wykonania na różne procesory**
- **Loop scalarization – zmiana wektorów/macierzy na skalary**
- **Loop fusion – scalenie zagnieżdżonych pętli**

## Testowanie zależności

- Wykrywanie zależności występujących w pętli pomiędzy iteracjami i instrukcjami
- Jest kilka możliwych rodzajów zależności pomiędzy instrukcjami
- Na ich podstawie buduje się graf zależności

```

for i = 2 to 4
  for j = 2 to 4
1:   a[i,j] = a[i-1,j+1];
  
```





AGH

## Istota badania zależności

```
DO i1 = L1, U1
  DO i2 = L2, U2
    ...
    DO in = Ln, Un
      A(f(i1, ..., in)) = ...
      ... = A(g(i1, ..., in))
    ENDDO
  ...
ENDDO
S1
S2
ENDDO
```



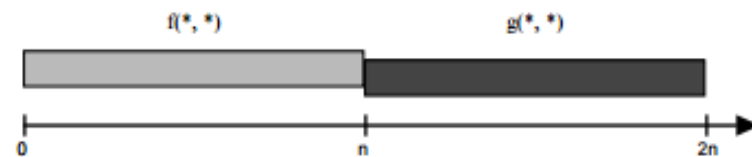
$$f(x_1, x_2, \dots, x_n) = g(y_1, y_2, \dots, y_n)$$

$$L_i \leq x_i, y_i \leq U_i, \forall i, 1 \leq i \leq n$$

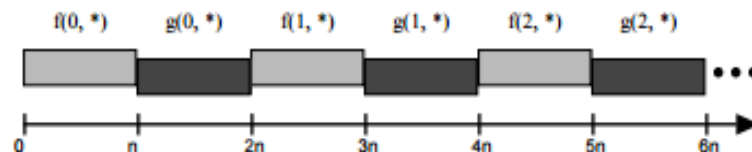
- Problem jest w gruncie rzeczy taki sam jak problem programowania całkowitoliczbowego
- Zbadanie zależności trochę utrudnia fakt, że to problem NP-zupełny

# Range Test

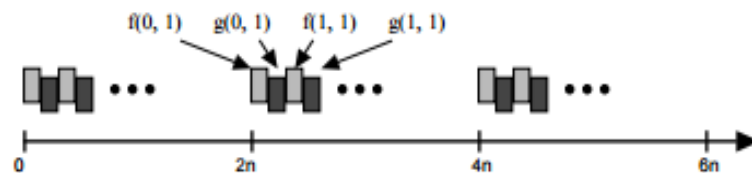
- Metoda na testowanie zależności
- Wykrywanie czy pomiędzy iteracjami nastąpiło odwołanie do tych samych elementów tablicy
- Wykorzystuje własności funkcji odwołującej się do tablicy



a)  $f(i, j) = i$ ,  $g(i, j) = i + n$



b)  $f(i, j) = 2*n*i + j$ ,  $g(i, j) = 2*n*i + j + n$



c)  $f(i, j) = 2*i + 2*n*j$ ,  $g(i, j) = 2*i + 2*n*j + 1$

## GCD test

$$f(x_1, x_2, \dots, x_n) = g(y_1, y_2, \dots, y_n)$$

$$f(x_1, x_2, \dots, x_n) = a_0 + a_1x_1 + \dots + a_nx_n$$

$$g(y_1, y_2, \dots, y_n) = b_0 + b_1y_1 + \dots + b_ny_n$$

$$a_0 - b_0 + a_1x_1 - b_1y_1 + \dots + a_nx_n - b_ny_n = 0$$

**-> równanie diofantyczne ma rozwiązanie wtw gdy:**

$$\gcd(a_1, \dots, a_n, b_1, \dots, b_n) \text{ divides } b_0 - a_0.$$

# Ohmega Test

- **Metoda Integer Programming**
- **Podobna do GCD test**
- **Wykonujemy przekształceń zmniejszających współczynniki by w końcu któryś z nich miał wartość 1 i mógł zostać wyeliminowany**

$$\begin{array}{l}
 7x + 12y + 31z = 17 \\
 3x + 5y + 14z = 7 \\
 1 \leq x \leq 40 \\
 -50 \leq y \leq 50
 \end{array}$$



$$0 \leq \sigma \leq 2$$



$$-|a_k|\sigma + \sum_{i \in V - \{k\}} \left( \left( \left\lfloor \frac{a_i}{m} + \frac{1}{2} \right\rfloor + (a_i \bmod m) \right) x_i \right) = 0$$

- **Poza jednym współczynnikiem pozostałe zmniejszyły się maksymalnie o 2/3**



## **Przykłady zastosowania testowania zależności**

- **Jeżeli pomiędzy iteracjami pętli nie ma zależności mogą być wykonane jednocześnie !!**
- **Jeżeli kierunki wektorów zależności są w porządku leksykograficznym po zmianie kolejności pętli to była ona poprawna**

## Istniejące rozwiązania

- **Yucca** – zmienia kod sekwencyjny w C na równoległy przy pomocy pthreads i OpenMP
- **PLUTO** – podobny do Yucca, zmienia C na równoległy z użyciem OpenMP(jest Open Source)
- **Par4All** – narzędzie do adaptacji kodu w C na wykonywany na różnych architekturach jak wielordzeniowe systemy, GPU
- **Cetus** – napisany w Javie, kilka strategii zrównoleglania kodu w C do OpenMP(jest Open Source)
- **Intel C++ Compiler** – wbudowane auto-parallelization zmieniające kod sekwencyjny na wykonywany wątkach

## Obecny stan prac

- **Generalny research literatury**
- **Przekształcenie kodu w C do postaci drzewa rozbioru**
- **Przekształcenie drzewa rozbioru do AST**
- **Rozpoczęcie prac nad generacją drzewa zależności**
- **Przygotowanie systemu do podpięcia istniejących algorytmów testowania zależności**

Pytania?