PaP

Projet : rapport2

4TIN804U

BERASATEGUY Tanguy, GOEDEFROIT Charles

Table des matières

1 ILP optimization (4.1)

On as fait les modifications :

Pour *ssandPile_do_ tile_ opt()* on a retirer les appelles à *table(out, i, j) pour passer par une varaible intermedier result. Cette modification petmer au compilateur de vectoriser car il peut maintenet facilement voir que les différante lige peuve être calculer en paralelle.*

```
int ssandPile_do_tile_opt(int x, int y, int width, int height)
{
  int diff = 0;

  for (int i = y; i < y + height; i++)
    for (int j = x; j < x + width; j++)
    {
-       table(out, i, j) = table(in, i, j) % 4;
+       int result = table(in, i, j) % 4;
-       table(out, i, j) += table(in, i + 1, j) / 4;
+       result += table(in, i + 1, j) / 4;
-       table(out, i, j) += table(in, i - 1, j) / 4;
+       result += table(in, i - 1, j) / 4;
-       table(out, i, j) += table(in, i, j + 1) / 4;
+       result += table(in, i, j + 1) / 4;
-       table(out, i, j) += table(in, i, j - 1) / 4;
+       result += table(in, i, j - 1) / 4;
-       if (table(out, i, j) >= 4)
+       table(out, i, j) = result;
+       if (result >= 4)
         diff = 1;
    }

  return diff;
}
```

```
int ssandPile_do_tile_opt(int x, int y, int width, int height)
{
  int diff = 0;

  for (int i = y; i < y + height; i++)
    for (int j = x; j < x + width; j++)
    {
      int result = table(in, i, j) % 4;
      result += table(in, i + 1, j) / 4;
      result += table(in, i - 1, j) / 4;
      result += table(in, i, j + 1) / 4;
      result += table(in, i, j - 1) / 4;
      table(out, i, j) = result;
      if (result >= 4)
```

```
41          diff = 1;
42      }
43
44    return diff;
45  }
```

Le gain de performance est de 2.37 car $\frac{178812}{75408}$

Pour asandPile_do_tile_opt() on a retirer...

```
1   int asandPile_do_tile_default(int x, int y, int width, int height)
2   {
3     int change = 0;
4
5     for (int i = y; i < y + height; i++)
6       for (int j = x; j < x + width; j++)
7   -       if (atable(i, j) >= 4)
8   +       int result = atable(i, j);
9   +       if (result >= 4)
10        {
11  +       result/=4;
12  -       atable(i, j - 1) += atable(i, j) / 4;
13  +       atable(i, j - 1) += result;
14  -       atable(i, j + 1) += atable(i, j) / 4;
15  +       atable(i, j + 1) += result;
16  -       atable(i - 1, j) += atable(i, j) / 4;
17  +       atable(i - 1, j) += result;
18  -       atable(i + 1, j) += atable(i, j) / 4;
19  +       atable(i + 1, j) += result;
20          atable(i, j) %= 4;
21          change = 1;
22        }
23    return change;
24  }
```

```
26  int asandPile_do_tile_opt(int x, int y, int width, int height)
27  {
28    int change = 0;
29
30    for (int i = y; i < y + height; i++)
31      for (int j = x; j < x + width; j++)
32      {
33        int result = atable(i, j);
34        if (result >= 4)
35        {
36          result/=4;
```

```
37          atable(i, j - 1) += result;
38          atable(i, j + 1) += result;
39          atable(i - 1, j) += result;
40          atable(i + 1, j) += result;
41          atable(i, j) %= 4;
42          change = 1;
43        }
44      }
45    return change;
46  }
```

Le gain de performance est de 1.2 car $\frac{37990}{31405}$

2 OpenMP implementation of the synchronous version (4.2)

3 OpenMP implementation of the asynchronous version (4.3)

4 Lazy OpenMP implementations (4.4)