用 CUDA 進行局部序列比對

Semi Global Sequence Alignment with CUDA

組員 吳信葆

指導教授 賀保羅

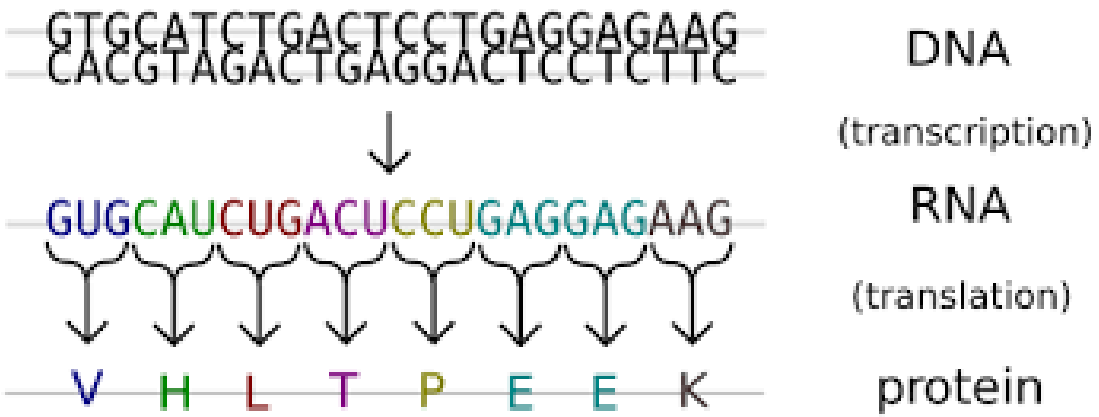
目錄

1) 序列比對
    1) 生物學背景
    2) Dynamic Programing
    3) CUDA
2) 演算法
    1) Semi Global Interval
    2) Alignment
3) 結果分析

目錄

1) 序列比對
    1) 生物學背景
    2) Dynamic Programing
    3) CUDA

轉錄、轉譯




```
int main(int argc,char** argv){
    byte *gx_int,*gy_int;
    afg_unit *M,*M1,*M2,*GM,*GM1,*GM2;
    int xsize,ysize;
    int nthread,nblock;
```

突變

a.DNA → segment fault ☹

b.DNA → compile error ☹

c.DNA → segment fault ☹

d.DNA → overflow ☹

e.DNA → better program ☺

序列比對

ATGCCATTCGT

ATTCCCGT

Gap Open=-3

Gap Extension=-2

1 1-2 1 1-3-2-2 1 1 1

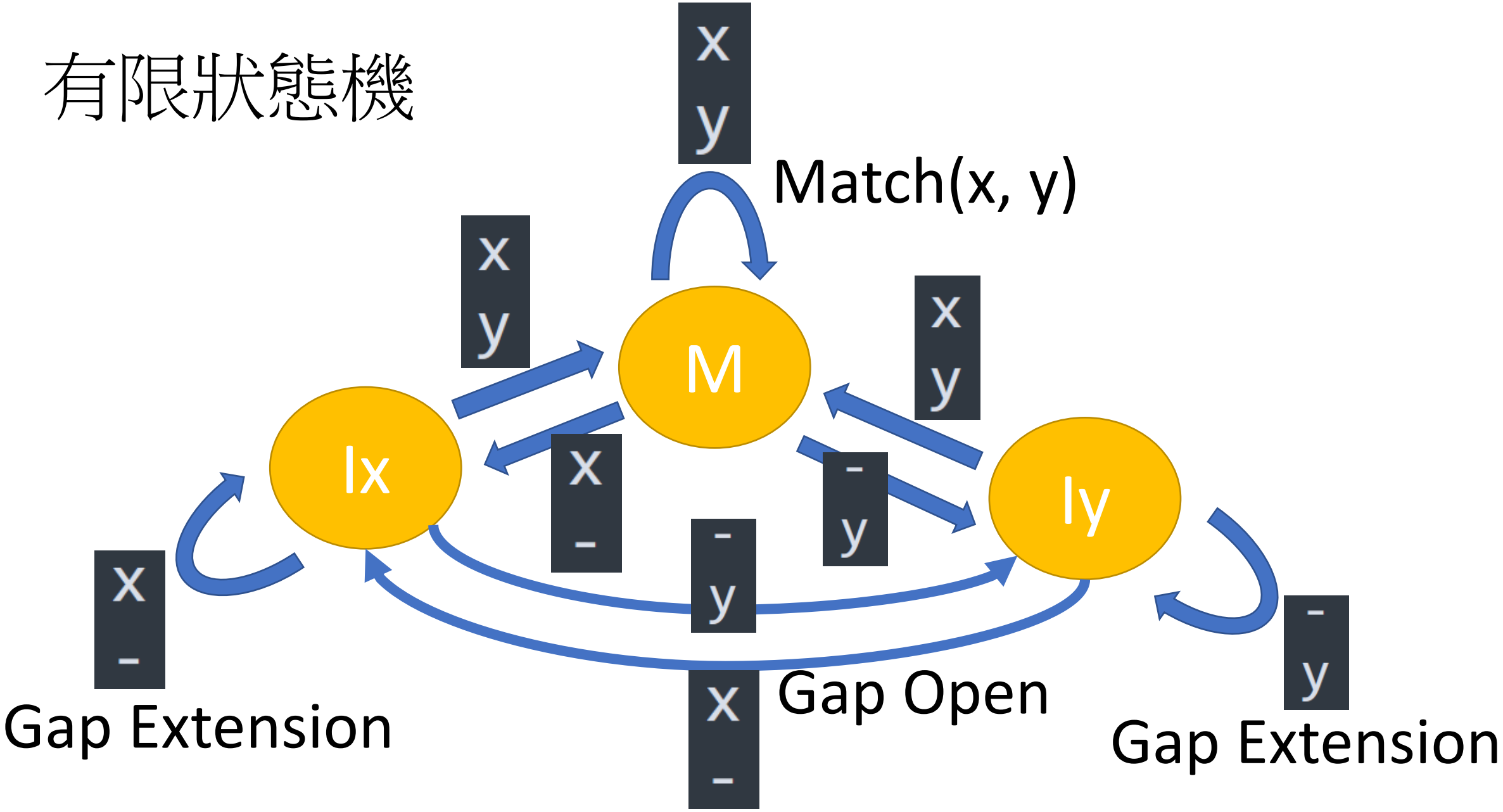
ATGCCATTCGT

ATTCC---CGT

=-2

	A	T	C	G
A	1	-2	-2	-1
T	-2	1	-1	-2
C	-2	-1	1	-2
G	-1	-2	-2	1

有限狀態機

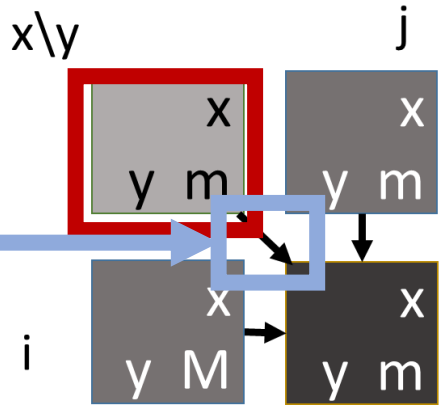

x
y
Match(x, y)
x
y
M
x
y
Ix
x
–
-
y
Iy
x
–
-
y
-
y
x
–
Gap Extension
Gap Open
Gap Extension

Dynamic Programming

$$\mathrm{M}(i,j) = match(Xi, Yj) + max \begin{cases} M(i-1,j-1) \\ Ix(i-1,j-1) \\ Iy(i-1,j-1) \end{cases}$$

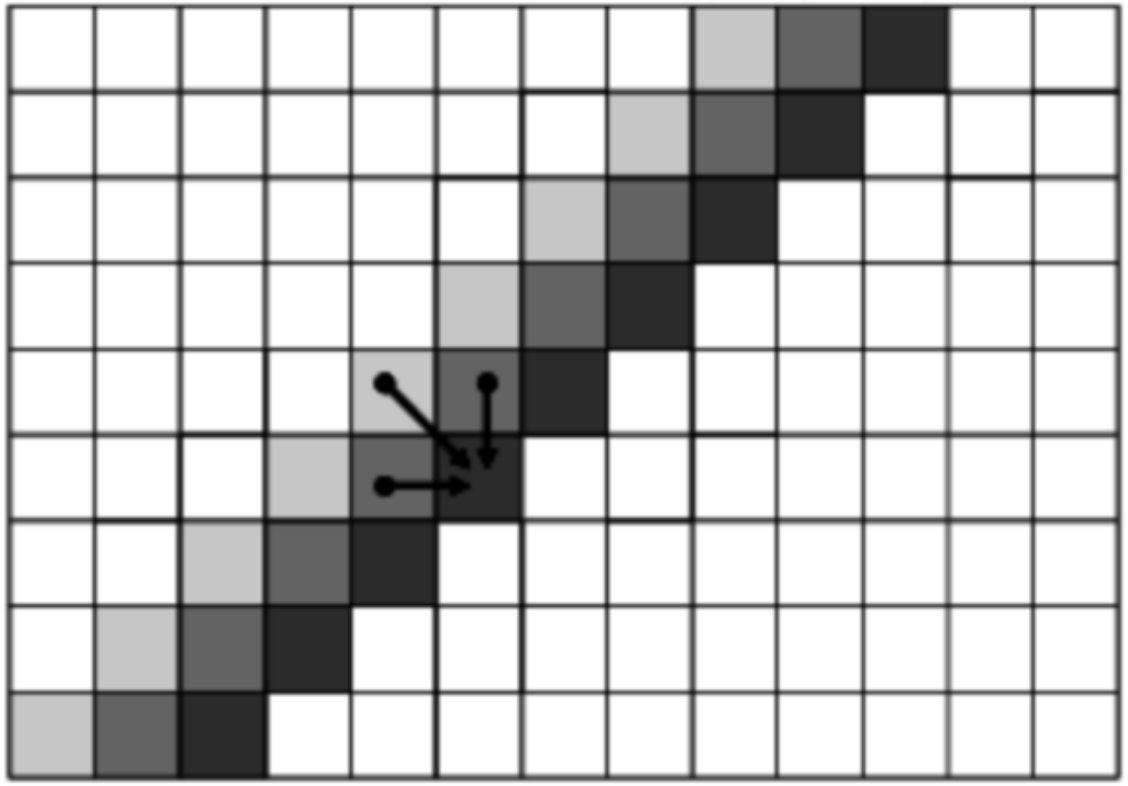
$$\mathrm{Ix}(i,j) = max \begin{cases} M(i-1,j) + gap \\ Ix(i-1,j) + ext \\ Iy(i-1,j) + gap \end{cases}$$

$$\mathrm{Iy}(i,j) = max \begin{cases} M(i,j-1) + gap \\ Ix(i,j-1) + gap \\ Iy(i,j-1) + ext \end{cases}$$



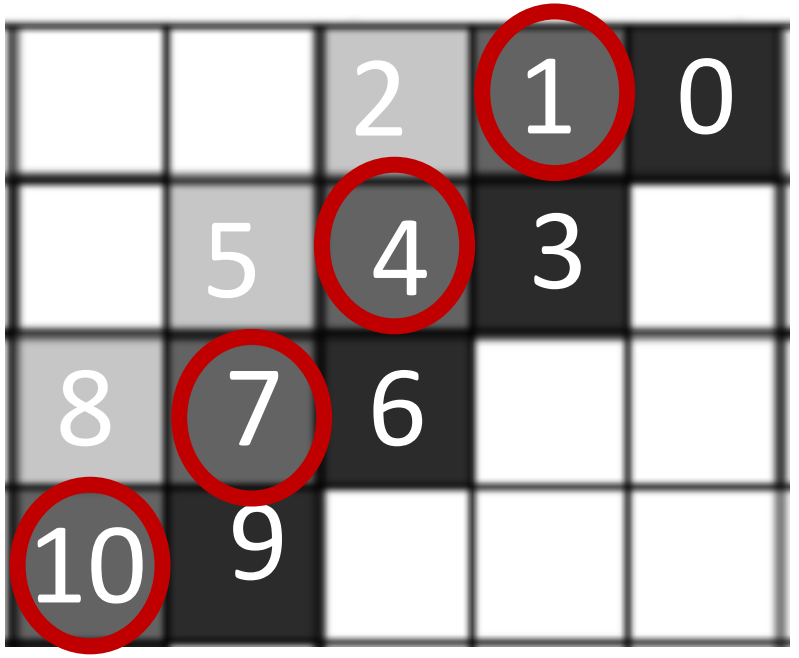
Data Dependency

x\y M2 M1 M



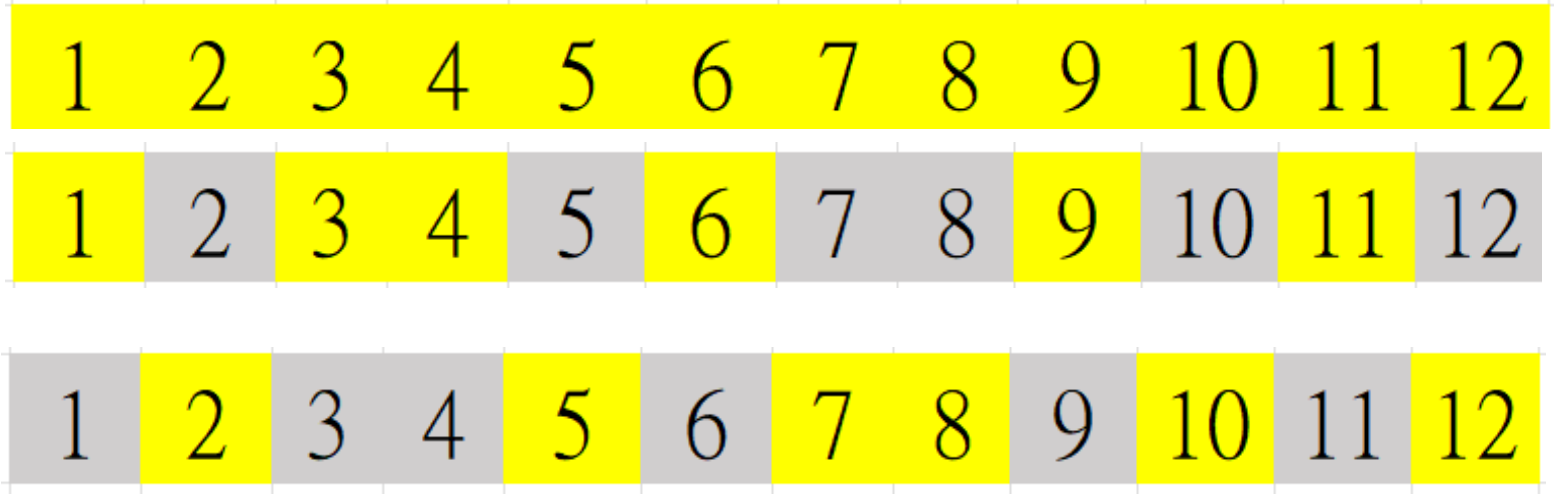
- 平行運算
- 線性空間複雜度

Coalescing

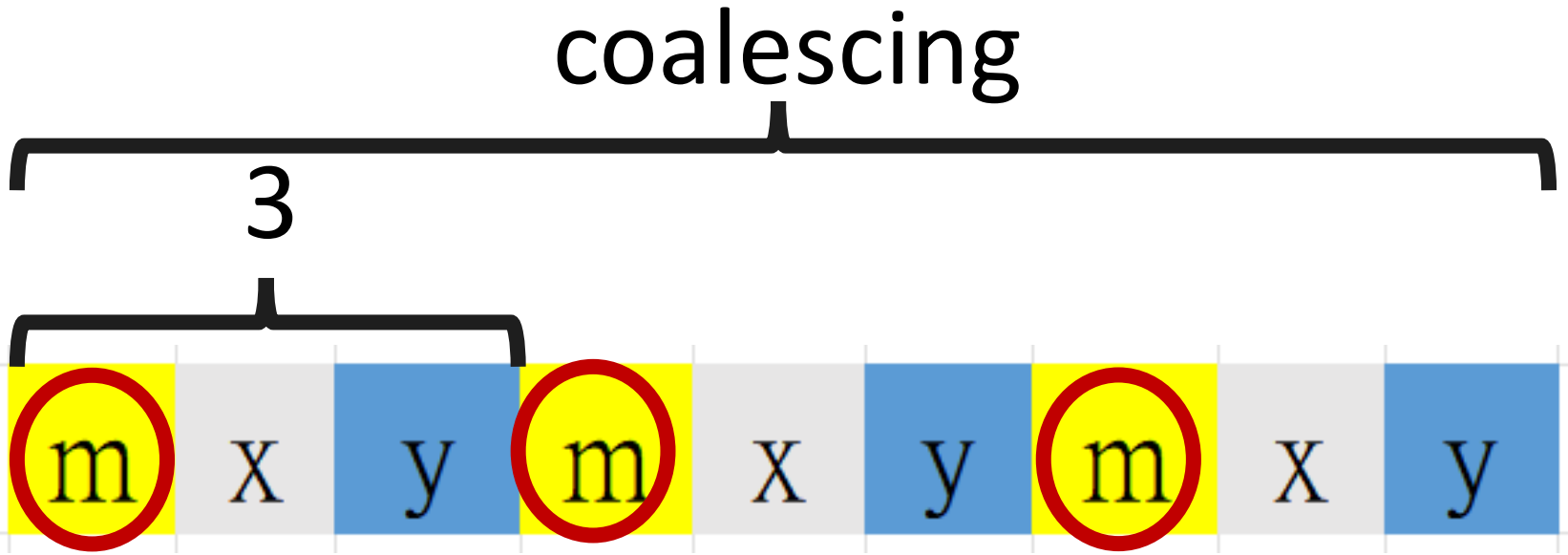


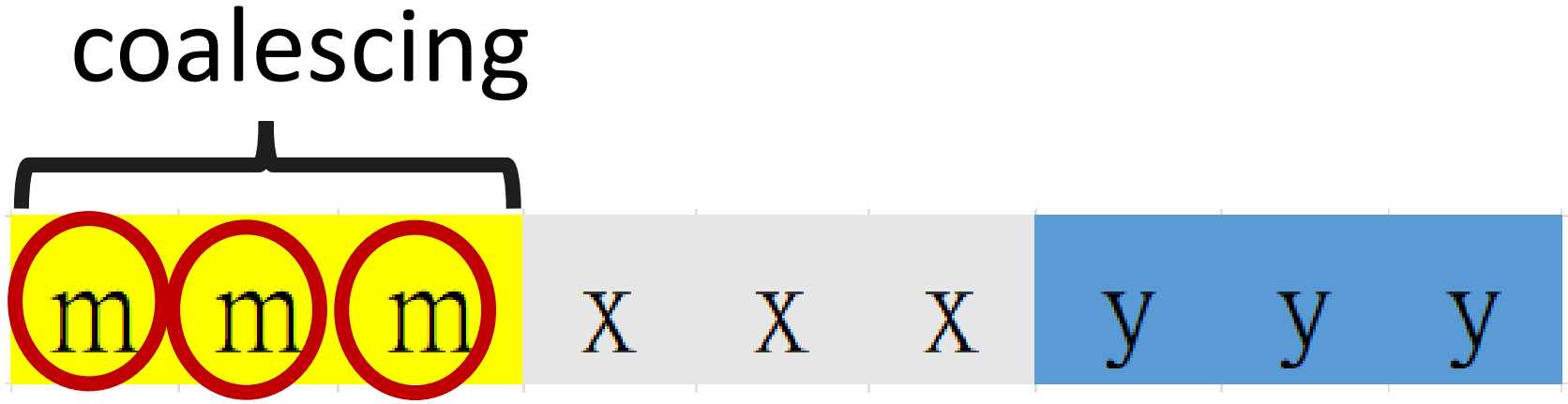
Global Memory: GPU 可存取的記憶體的其中一種，
因為 CUDA Thread 只要有指標皆可存取，故命名之。
SIMD: single instruction multiple data

```
if(a[tid]>5){
    c=a[tid];
}else{
    c=b[tid];
}
```



Warp: 在一個 Block 中連續的 threads，通常是 32 條。
Block: GPU 會把多條連續的 threads 分成多個 Blocks，每個 Block 具有一個 Multiprocessor
Multiprocessor: 平行運算單位，他會挑選一個 Warp 的 threads 進行 SIMD 平行運算


coalescing
3
m
x
y
m
x
y
m
x
y


coalescing
m
m
m
x
x
x
y
y
y

Above CUDA 3.0

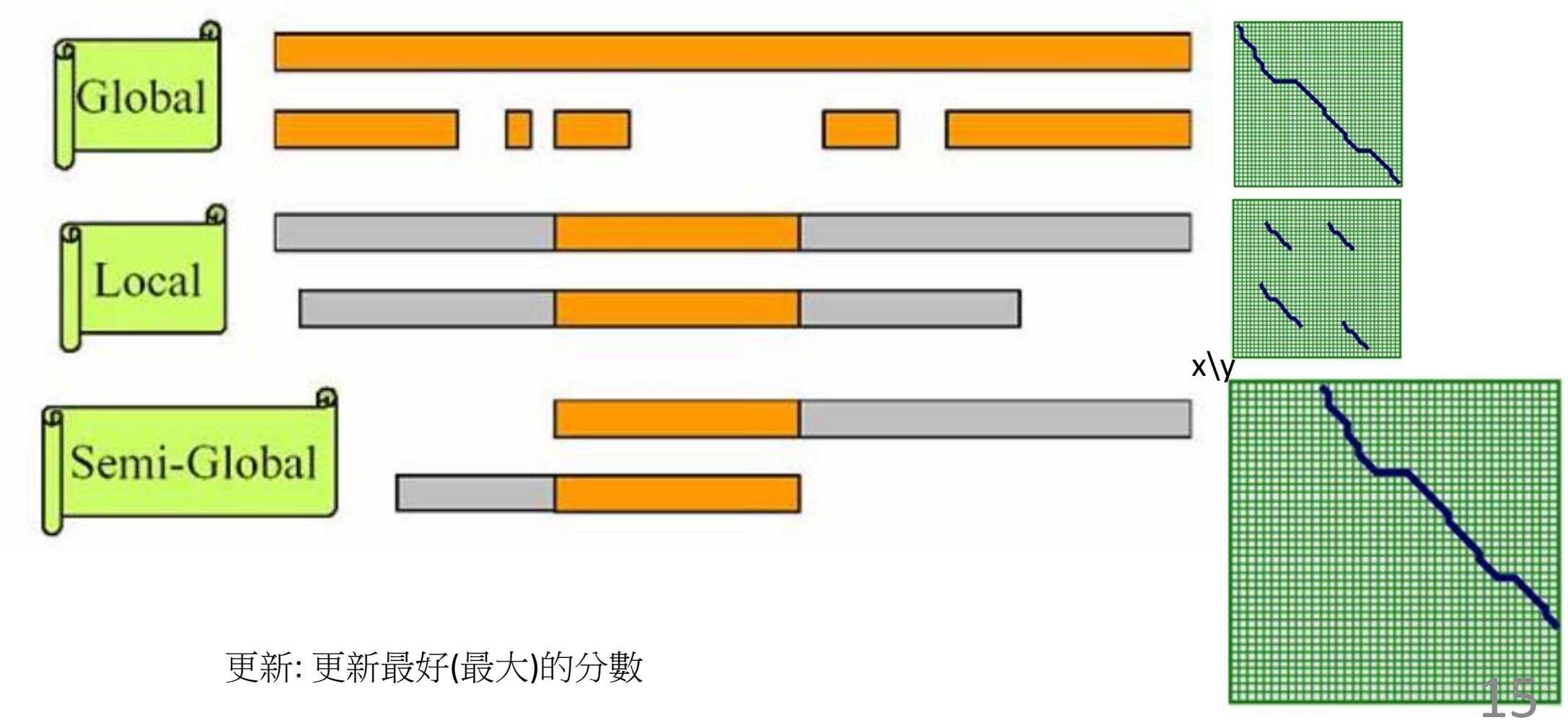
Normal code 😐 ⟶ good performance ☺

excellent code 😎 ⟶ excellent performance 😎

目錄

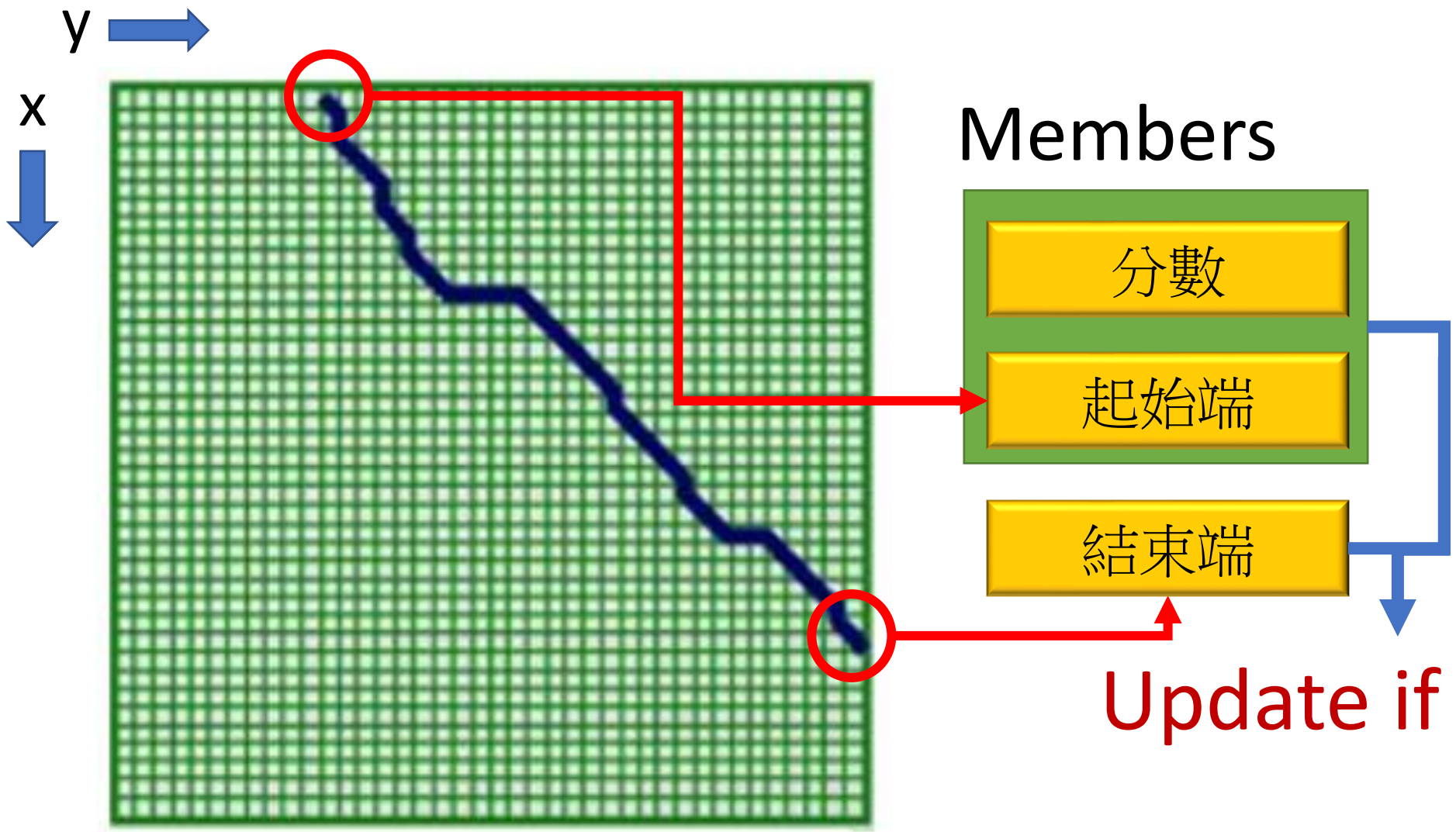
1) 序列比對
    1) 生物學背景
    2) Dynamic Programing
    3) CUDA
2) 演算法
    1) Semi Global Interval

Semi Global

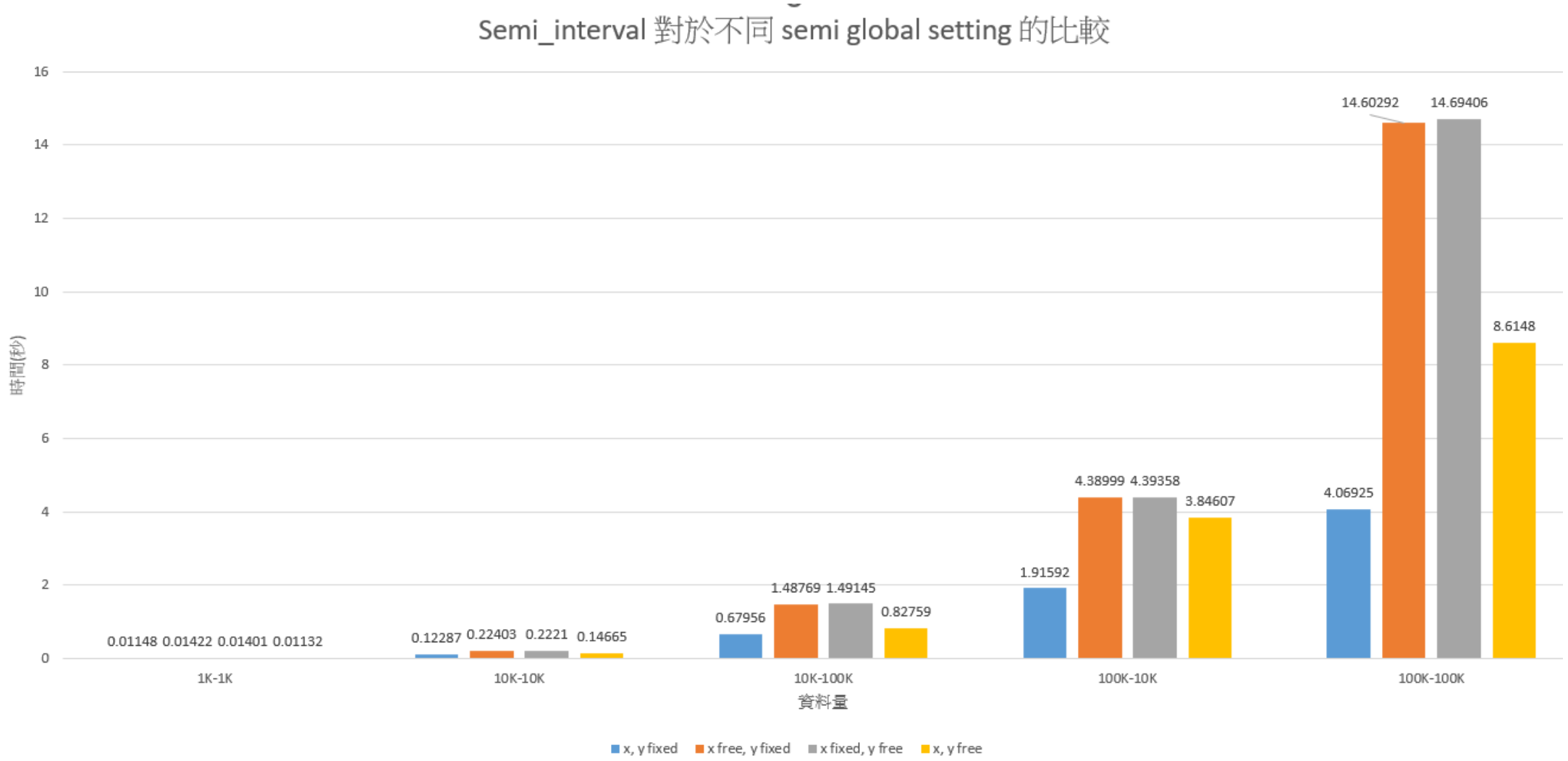


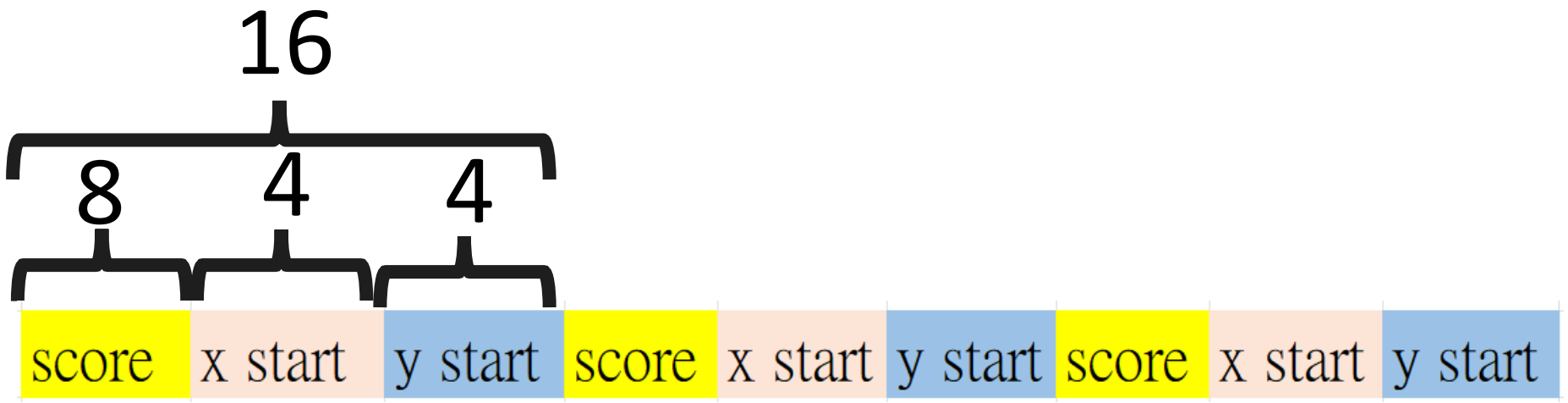
更新: 更新最好(最大)的分數

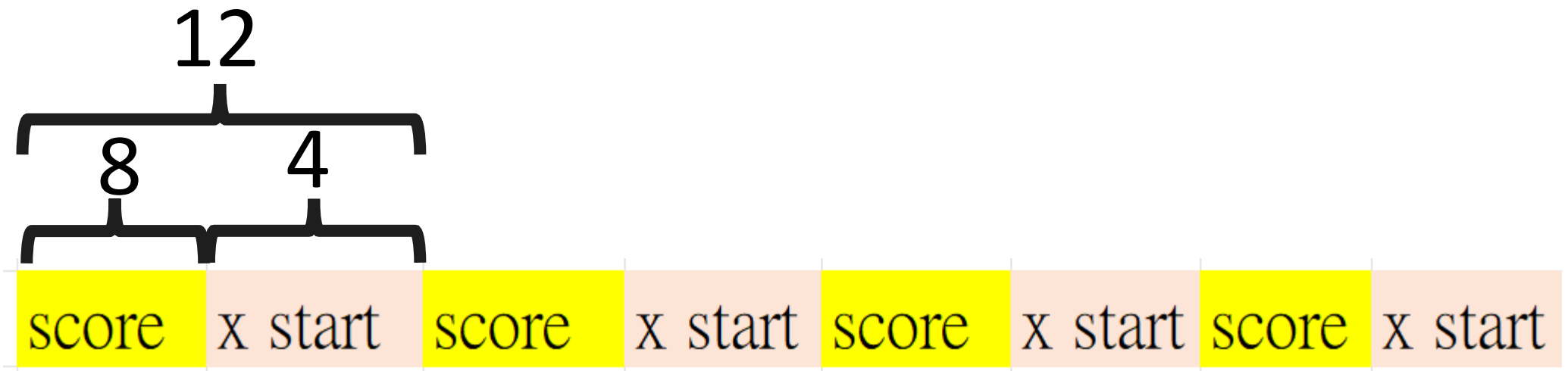
取得比對區間



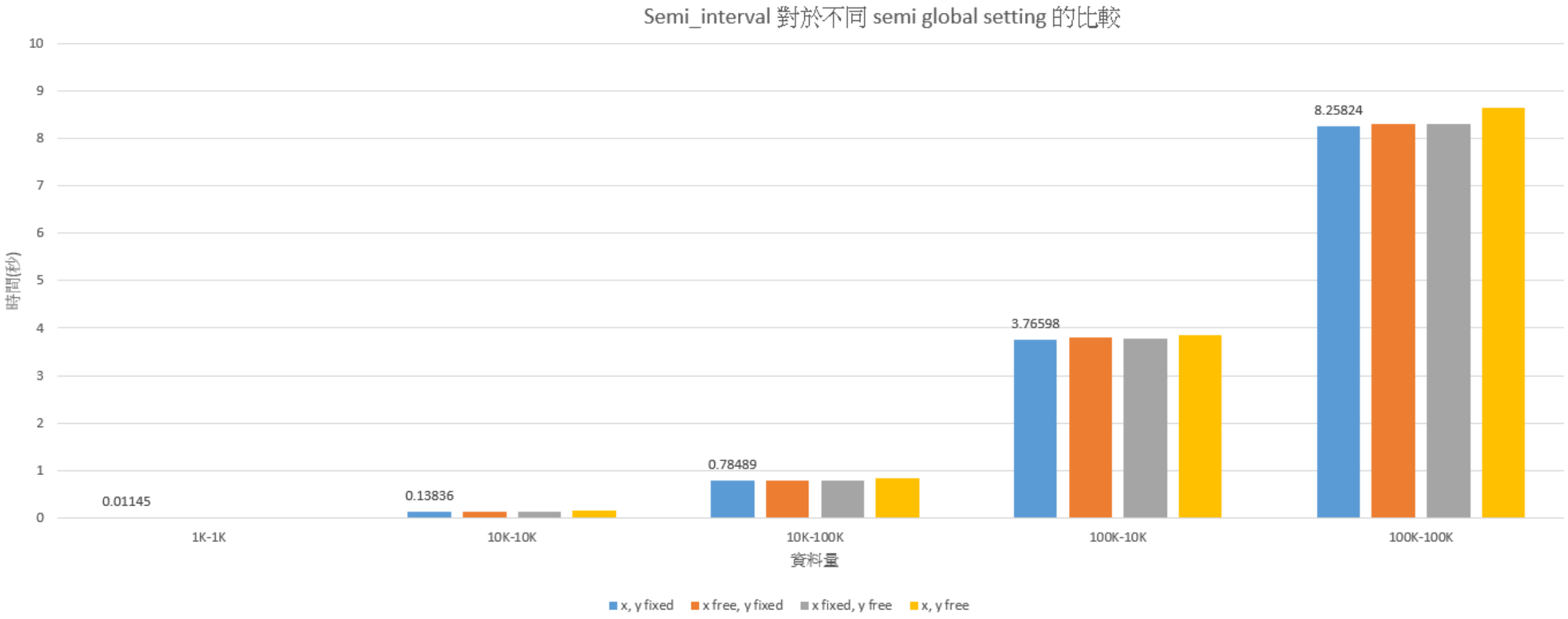
Semi Global Setting




16
8
4
4
score
x start
y start
score
x start
y start
score
x start
y start


12
8
4
score
x start
score
x start
score
x start
score
x start

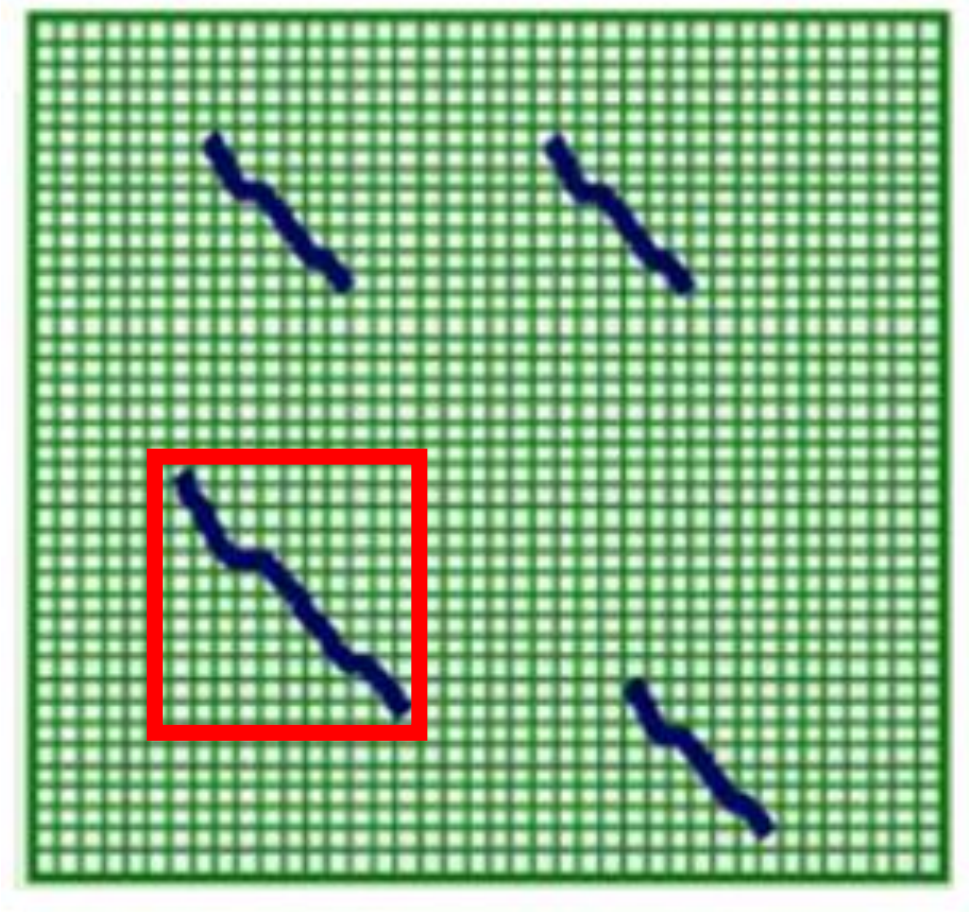
Semi Global Setting (記憶體對齊後)



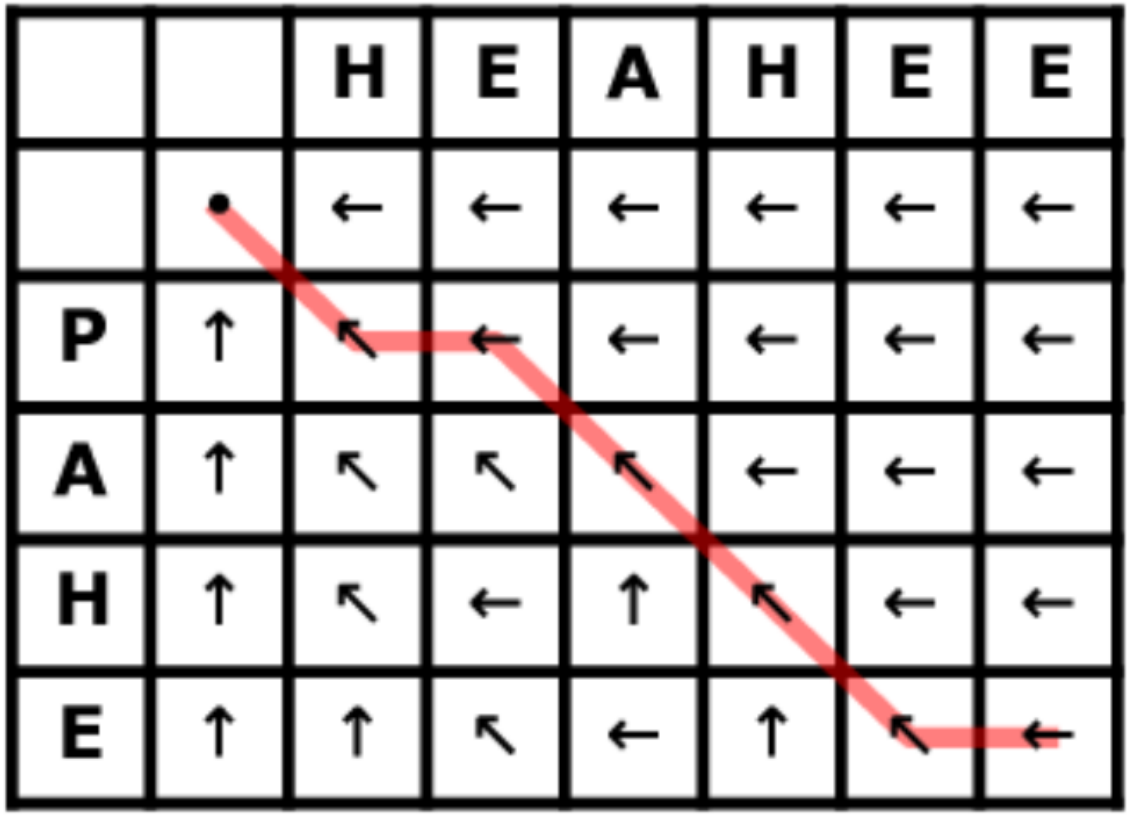
目錄

1) 序列比對
    1) 生物學背景
    2) Dynamic Programing
    3) CUDA
2) 演算法
    1) Semi Global Interval
    2) Alignment

如何得到序列比對

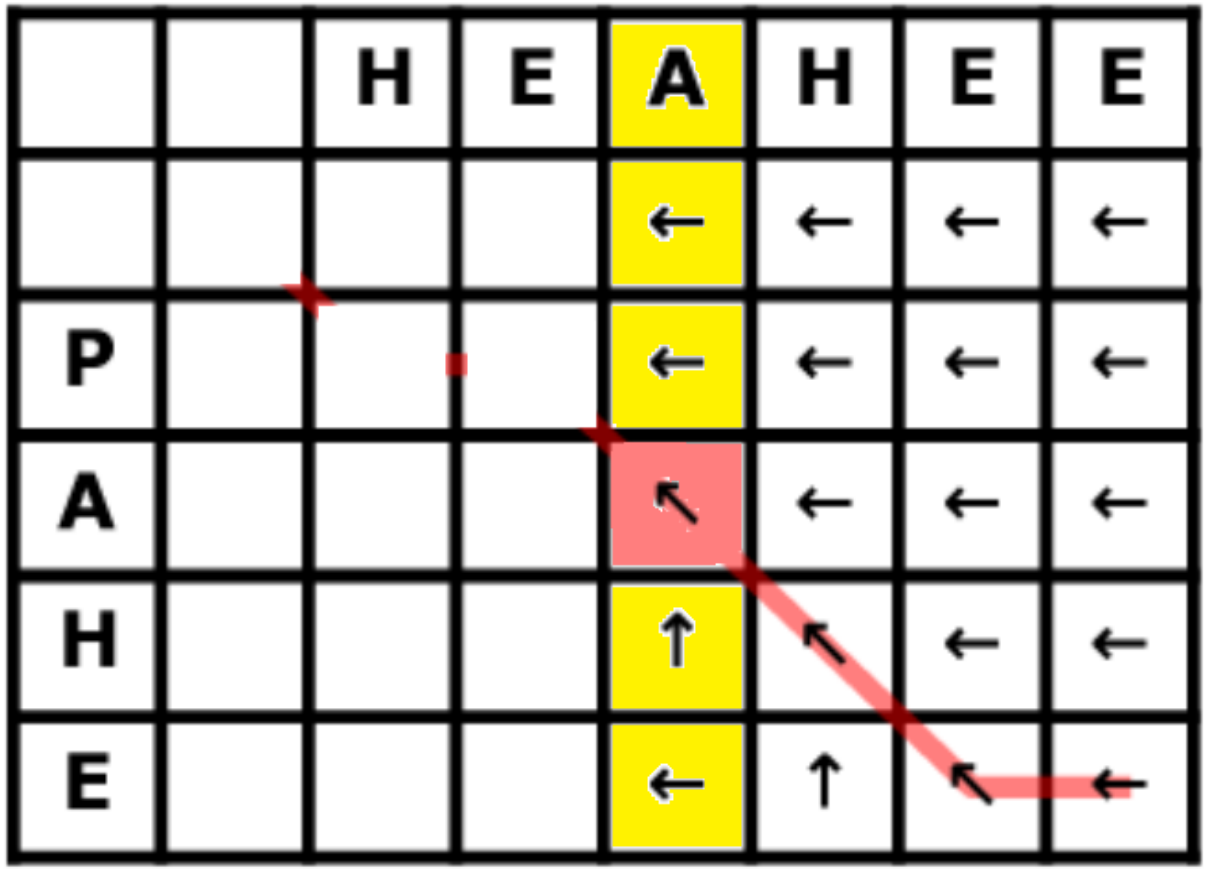


產生比對

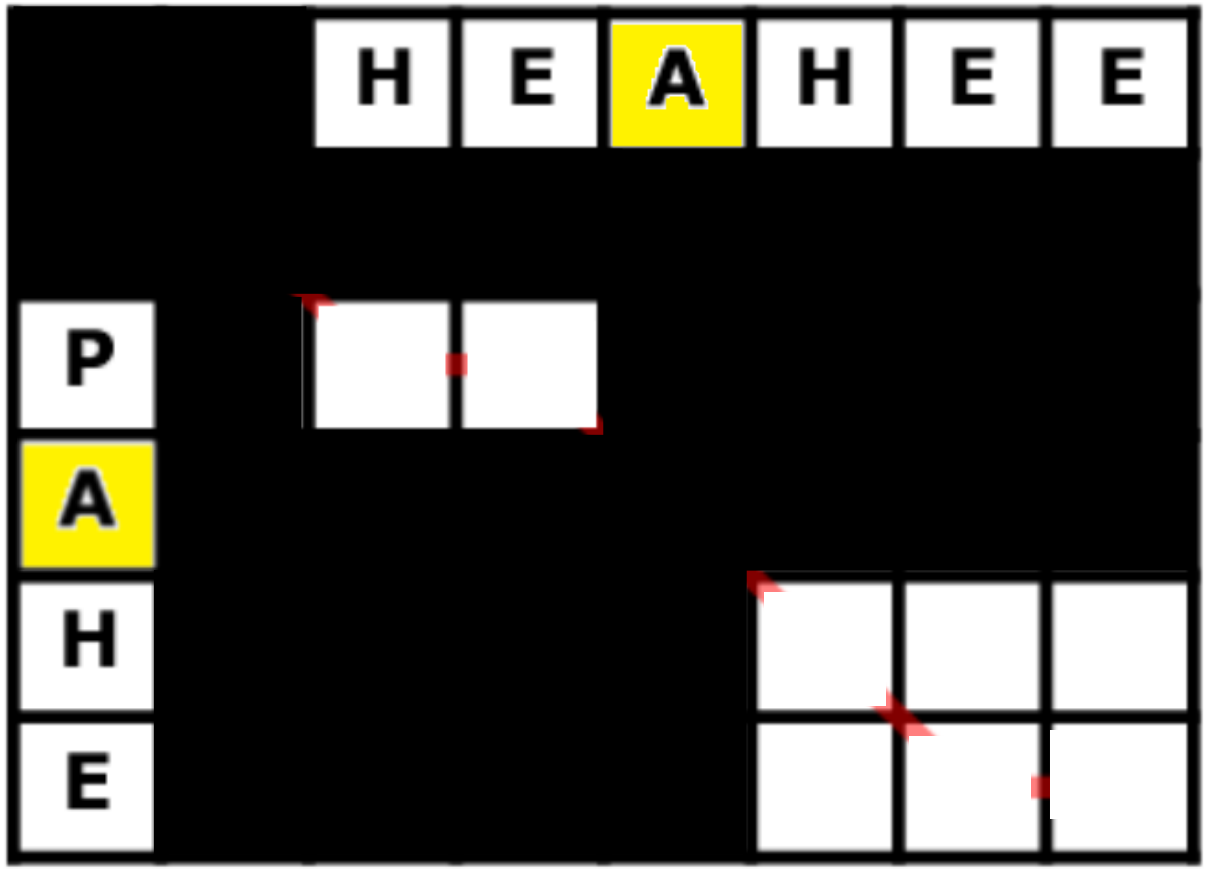


空間複雜度
=O(n^2)

找中間點



切割

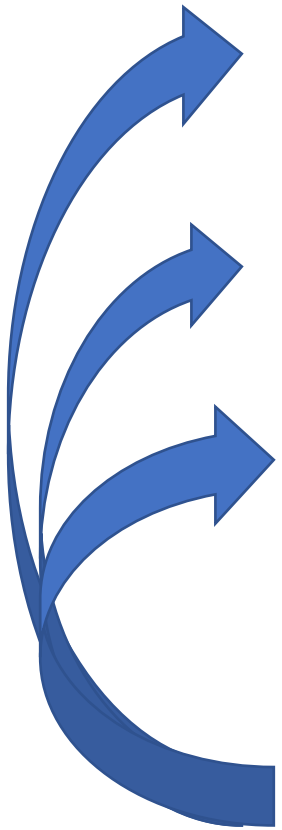


T(n,m)

=nm+nm/2+nm/4+...

=O(nm)

Conquer and Divide



1. 遞迴中斷點可使用O(n^2)演算法 (60000*60000) (印出比對結果)。
2. 找到切割點，將題目分成兩個子問題，(0,0) 初始化在函式外。
3. 解決左邊的子問題
4. 印出切割點比對，若 x 是一個 gap，右邊子問題 (0,0) 位置初始為 (m,x,y):=(-inf,-inf,0)，否則為 (0,-inf,-inf)
5. 解決右邊的子問題

目錄

1) 序列比對
   1) 生物學背景
   2) Dynamic Programing
   3) CUDA
2) 演算法
   1) Semi Global Interval (Dynamic Programing)
      - Dynamic Programing with CUDA
   2) Alignment (Conquer and Divide)
3) 結果分析

時間複雜度

n>>t, n, m 是 X, Y 序列的長度，t 為 GPU 可平行化數量的最大值

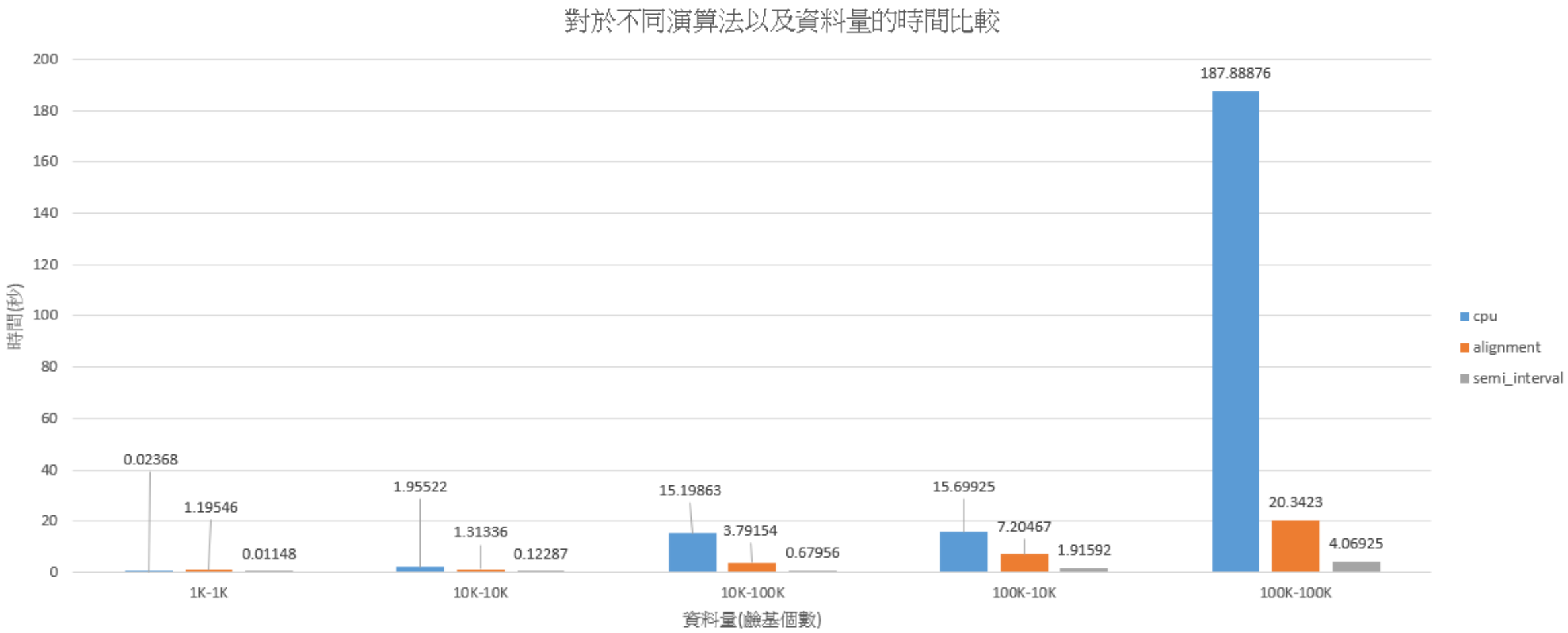
Semi Global Interval

- CPU: T(n,m)=O(nm)
- GPU: T(n,m)=O((n/t)m)

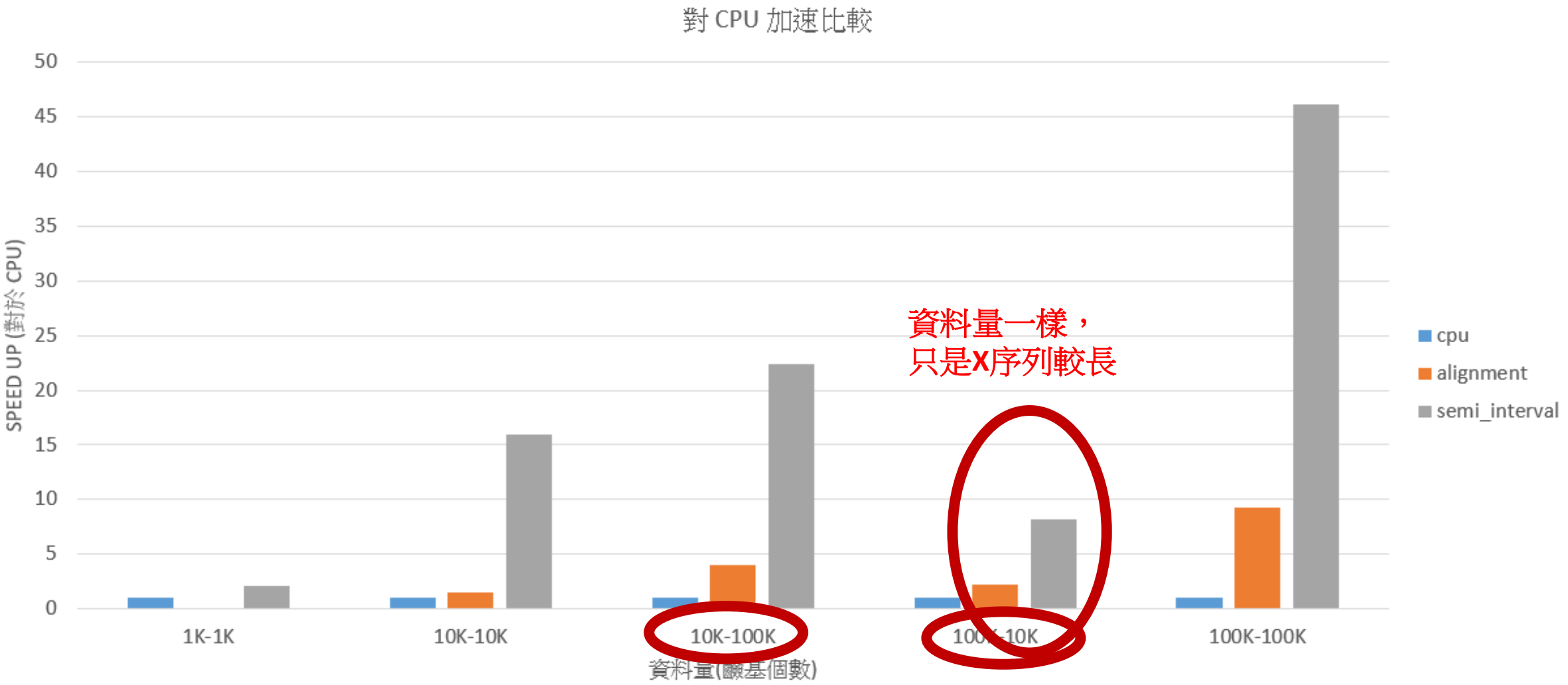
Alignment

- CPU: T(n,m)=O(nm)
- GPU: T(n,m)=O((n/t)m)　　;if t==n (O(mlgn))

Global alignment

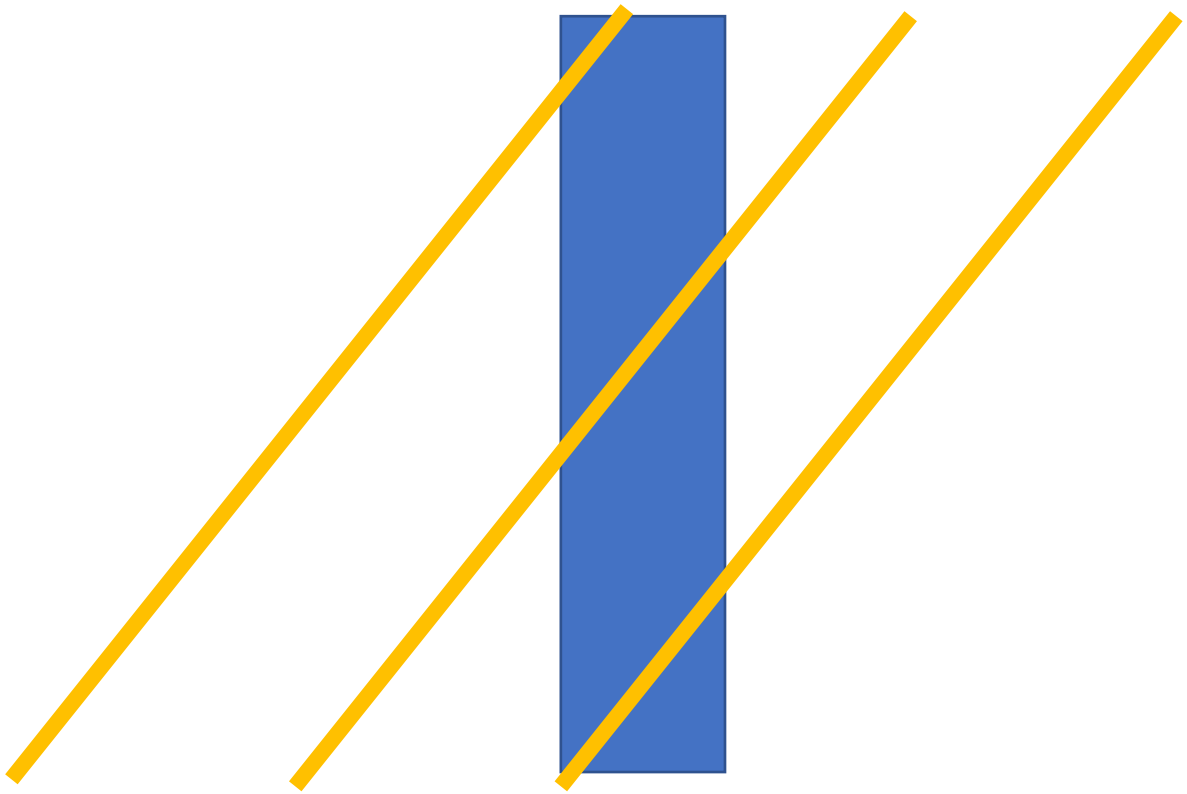


Global alignment



Cache Miss

10K-100K



100K-10K

我的原始碼

- https://github.com/paul90317/Semi-Sequence-Alignment-with-Cuda

致謝

賀保羅 教授(指導教授)

盧宥霖 學長(實驗室裡做 CUDA 研究的學長)

圖片 連結

- Data Dependency
- Semi Global
- Semi Global Matrix