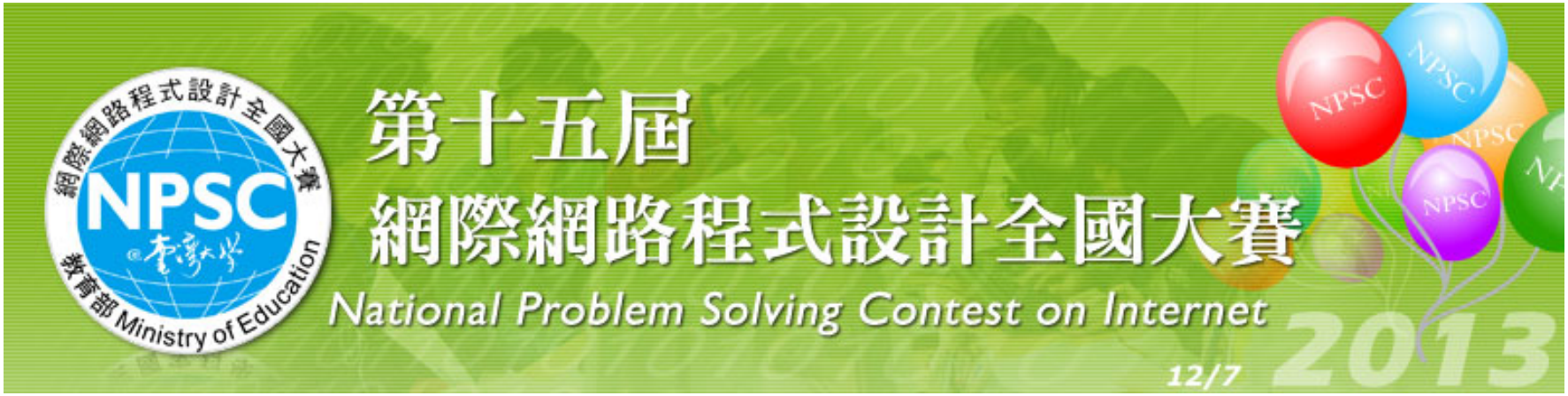


高中組決賽

December 7, 2013

題目 A. 可魚果運輸問題

題目

- 在一張圖上找 F 條路徑，使總 cost 最小。
- 每條邊一但使用超過某個次數 cost 就會改變。

題目 A. 可魚果運輸問題 (cont.)

概況

- 第一位通過 (First Blood)：team4-板橋高中 01，26 分鐘。
- 通過隊數：21

題目 A. 可魚果運輸問題 (cont.)

解題說明

- 直覺：找很多條路徑 + 小小張的圖 ($N \leq 100$) ？
- 哈哈！網路流吧！複雜度好像很對！

題目 A. 可魚果運輸問題 (cont.)

- 膝蓋中箭：範測不會過 OAO.
- $\forall C' \leq C$ ，跟花費流正確的條件反過來 QQ

題目 A. 可魚果運輸問題 (cont.)

- 觀察：你不會想要分多條路走 OAO ！
- 任兩條路 A, B ，把 A 換成 B 或把 B 換成 A ，至少有一個 cost 不會上升。

題目 A. 可魚果運輸問題 (cont.)

- **直觀證明 1：同一條路只會越用越便宜**
- cost(A) $\geq$ cost(**多加一條** A)
- **直觀證明 2：若** A **換成** B cost **上升**
- cost(B) $\geq$ cost(**加** B) $>$ cost(A) $\geq$ cost(**加** A)
- B **換成** A **會降** cost 。

題目 A. 可魚果運輸問題 (cont.)

- 所以：一條邊要嘛就不用，要嘛就用 F 次。
- Sol：每條邊的 cost 設為走 F 次時的花費。
- 耶！變成一般的最短路！

題目 A. 可魚果運輸問題 (cont.)

Yes

題目 A. 可魚果運輸問題 (cont.)

- Too boring?

題目 A. 可魚果運輸問題 (cont.)

- Let's do it with 古可魚語！

題目 B. 薑餅人的新樂園

題目

- 將 N 個整數分成 M 群，每一群的成本 G_i 為全距。
- 問所有群的 G_i 總和最小為多少。

題目 B. 薑餅人的新樂園 (cont.)

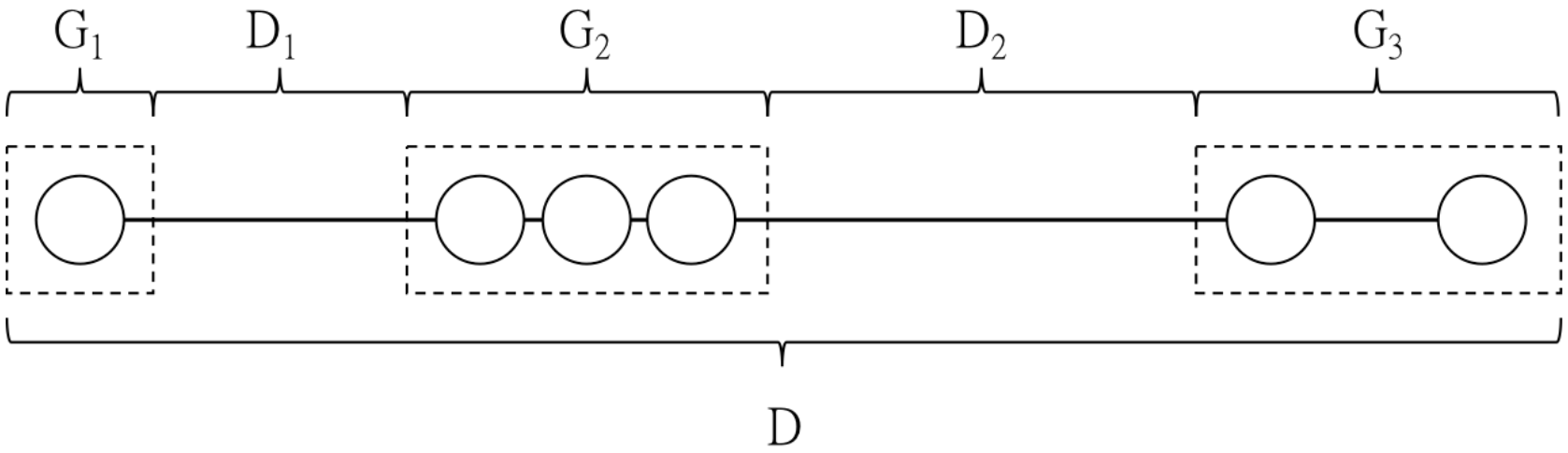
概況 (封版時)

- 第一位通過 (First Blood)：team15-實驗中學 01，22 分鐘。
- 通過隊數：27

題目 B. 薑餅人的新樂園 (cont.)

解題說明

- 任兩群的區間皆不重疊。
- 最小群全距和 $=$ 全距 $-$ 最大群間距和
  Example: $G_1 + G_2 + G_3 = D - (D_1 + D_2)$



題目 B. 薑餅人的新樂園 (cont.)

- 先對 N 個整數排序找出 $N-1$ 個間距。
- 再對間距排序後找出前 $N-M$ 大者取總和。

請用 $O(NlgN)$ 的排序！

題目 C. 棒球練習場

題目

- 對於一個 (x, y) 每一次操作都會變成 $(x \oplus y, |x - y|)$。
- 問 $[1, N] \times [1, M]$ 有幾對數字經過多次操作會變成 $(0, 0)$？

題目 C. 棒球練習場 (cont.)

概況 (封版時)

- 第一位通過 (First Blood)：team2-高雄中學 04，6 分鐘。
- 通過隊數：28 (全過)！

題目 C. 棒球練習場 (cont.)

解題說明

- 試過幾次之後，幾乎所有 (x, y) 都會變成 $(0, 0)$。
- 猜測答案為 $NM \rightarrow$ Yes !
- 理由：
  - 我們觀察 x, y 之中最低是 1 的 bit。
    $\rightarrow$ 至多兩次操作，可以使最低 bit 變成 0。
  - 令 k 為 x, y 的最高 bit。則 $x - y$ 和 $x \oplus y$ 的最高 bit 都不會超過 k。

題目 C. 棒球練習場 (cont.)

常見錯誤

- 請不要猜 lcm(N, M), XD。

題目 D. 樣式圖像辨識

題目

- 「樣式」：四方向、由 '*' 構成的連通塊。
- 基本樣式圖：$R_1 \times C_1$ 的圖像，其中恰包含一個樣式，該樣式稱為「基本樣式」。
- 文本：$R_2 \times C_2$ 的圖像，其中至少包含一個樣式。

- Q: 給一基本樣式圖和一目標文本，問目標文本中是否包含該基本樣式。

題目 D. 樣式圖像辨識 (cont.)

概況 (封板時)

- 第一位通過 (First Blood): None。
- 通過隊數: 0 隊。

題目 D. 樣式圖像辨識 (cont.)

解題說明

- 圖像大小為 $R_1 \times C_1$ 和 $R_2 \times C_2$。
- $1 \leq R_1, C_1, R_2, C_2 \leq 1024$。

- 要考慮「旋轉」怎麼辦？
  $\Longrightarrow$ 旋轉 $=$ 分開做四次。

題目 D. 樣式圖像辨識 (cont.)

- **暴力比對**：$O(R_1 \times C_1 \times R_2 \times C_2)$
  $\Longrightarrow$ No - Time Limit Exceeded。
- **目標**：$O(R_1 \times C_1 + R_2 \times C_2)$

題目 D. 樣式圖像辨識 (cont.)

- 第一步：找出基本樣式的「連通塊」。
- 每個位置至多被看到常數次 $\Longrightarrow O(R_1 \times C_1)$。
- 注意：找連通塊時如果使用遞迴版本 DFS
  $\Longrightarrow$ 遞迴過深 (No - Runtime Error)

題目 D. 樣式圖像辨識 (cont.)

- 第二步：找出目標文本中所有的連通塊。
- 同樣地，文本中的每個位置至多被看到常數次 $\Longrightarrow O(R_2 \times C_2)$。

題目 D. 樣式圖像辨識 (cont.)

- 第三步：對於每個文本中的連通塊，與基本樣式做比對。
- 這步的複雜度？均攤分析 (amortized analysis)！
- 每個的連通塊只會被比較一次 $\Rightarrow$ 所有的 * 最多只會被比較一次。
- 然而，文本中的 * 的數量最多只會有 $O(R_2 \times C_2)$ 個。
- 故此步的複雜度實際上為 $O(R_2 \times C_2)$。

題目 D. 樣式圖像辨識 (cont.)

- 總複雜度為以上個步驟之複雜度相加，故為

$$
\begin{aligned}
& O(4 \times (R_1 \times C_1 + R_2 \times C_2 + R_2 \times C_2)) \\
=& O(R_1 \times C_1 + R_2 \times C_2)
\end{aligned}
$$

$\Longrightarrow$ Yes

題目 E. 可魚果贈送問題

題目

- N 個數字 $C_1, C_2, \ldots, C_N$，問能不能挑出兩堆總和一樣。
- $N, C_i \leq 50000$。

題目 E. 可魚果贈送問題 (cont.)

概況 (封版時)

- 第一位通過 (First Blood)：team15-實驗中學 01，66 分鐘。
- 通過隊數：3

題目 E. 可魚果贈送問題 (cont.)

解題說明

- 50000^3 背包 OAQ? 不合理啊。
- 如果真的有五萬個東西 ...
- 隨便挑兩個有大約 50000^2 種
- 可是數字種類不會超過 10^5。
- 根本就一定是 Yes 嘛！

題目 E. 可魚果贈送問題 (cont.)

- 進一步分析，N 個東西總共有 $2^N - 1$ 種挑法。
- 總和通通落在 $50000 \times N$ 裡面。
- 也就是說如果 $2^N - 1 > 50000 \times N$ 就一定有解。
- `if (N >= 20)puts("Yes");`
- 如果不到 20 的話就就 2^N 暴搜即可。

題目 F. 可魚果滾動問題

題目

- 給你一個凸多邊形。
- 多邊形內部有一堆點，各有各的方向向量。
- 求最後掉出去的點走了多遠。

題目 F. 可魚果滾動問題 (cont.)

概況 (封版時)

- 第一位通過 (First Blood)：
  team3-成功高中 02，91 分鐘。
- 通過隊數：7

題目 F. 可魚果滾動問題 (cont.)

解題說明

- 顯然，每個點掉出去時走的距離就是其距離其與邊界的交點。
- 但是，我們不知道究竟交點在那。

題目 F. 可魚果滾動問題 (cont.)

- 題本上，$N \times M$ 小小的
- 對每個內部的點枚舉每一條多邊形的邊計算交點！

題目 F. 可魚果滾動問題 (cont.)

- Sol? 點斜式
- 每條線用起始點與斜率表示，亂解一陣交點！

題目 F. 可魚果滾動問題 (cont.)

No - Runtime Error

題目 F. 可魚果滾動問題 (cont.)

- 陷阱 1：垂直的東西斜率會咻蹦，特別判！

題目 F. 可魚果滾動問題 (cont.)

No - Wrong Answer

題目 F. 可魚果滾動問題 (cont.)

- 陷阱 2：平行的東西沒有交點，特別判！

題目 F. 可魚果滾動問題 (cont.)

No - Wrong Answer

題目 F. 可魚果滾動問題 (cont.)

- code 變一團垃圾、精度崩解。
- 反省：這不是個好 idea 。

題目 F. 可魚果滾動問題 (cont.)

- **Sol! 向量**
- **改用向量作，輕鬆各種** dot 、cross **！**

題目 F. 可魚果滾動問題 (cont.)

Yes

題目 F. 可魚果滾動問題 (cont.)

```
1  double calc(const vector &p, const vector &q) const {
2
3      vector pq = q - p;
4      if (dir.cross(pq) == 0) return 0; // parallel
5
6      vector pj =
7          pq * (pq.dot(pos - p) / pq.sqlen()) + p; // projection
8      vector foot = pj - pos;
9      double coef = foot.sqlen() / foot.dot(dir);
10     if (coef < 0) return 0; // never intersect
11
12     vector crs = dir * coef + pos;
13     if ((crs - p).dot(crs - q) > 0) return 0; // not on the seg
14     else return (crs - pos).length();
15 }
```

題目 F. 可魚果滾動問題 (cont.)

Extreme：其實這題可以出 $N, M = 100000$ ！

題目 G. 胖胖天大大薯 II

題目

- 胖胖天吃大薯（嚼嚼）。
- 接下來 N 天第 i 天可以不吃或吃 C_i 或吃 $2C_i$ 根。
- 吃的大薯數量要非遞減，已知上次胖胖天吃了 M 根。
- 問接下來 N 天胖胖天最多可以吃幾天大薯。

題目 G. 胖胖天大大薯 II (cont.)

概況 (封版時)

- **第一位通過** (First Blood)：team8-**師大附中** 02，33 **分鐘**。
- **通過隊數**：17

題目 G. 胖胖天大大薯 II (cont.)

解題說明

- 如果只能不吃或吃 C_i？
- $C_1, C_2, \ldots, C_N$ (不到 M 的丟掉) 的最長非遞減子序列。

題目 G. 胖胖天大大薯 II (cont.)

- 那 $2C_i$ 如何是好？
- $C_i, 2C_i$ 不能同時選到！
- $2C_1, C_1, 2C_2, C_2, \ldots, 2C_N, C_N$ (不到 M 的丟掉)的最長非遞減子序列。

題目 H. 古可魚語

題目

- 定義一種類似 Lisp 的古可魚語，要求你實作一個直譯器。

題目 H. 古可魚語 (cont.)

概況 (封版時)

- 第一位通過 (First Blood)：None
- 通過隊數：None

題目 H. 古可魚語 (cont.)

解題說明

- 時限有三十秒！速度不重要！重點是概念。

題目 H. 古可魚語 (cont.)

- 定義 Object 類，衍生出 Integer 與 Function
- 定義 Environment 類，用來將字串對應到 Object

題目 H. 古可魚語 (cont.)

- struct Object;
- typedef map<string, Object*> Env;
- struct Int : Object;
- struct Callable : Object;
- struct Func : Callable;
- struct NativeFunc : Callable;
- ... and so on.

題目 H. 古可魚語 (cont.)

- 照著 parse 遇到什麼就做什麼，不用預處理
- 注意 if 時要先判斷 condition ，只能作該走的那半
- 注意 lambda 時必須把當前環境存在 Object 內 (直接 copy 夠快)
- 注意 define 時該修改哪個 Environment ，否則遞迴之類的會有問題

題目 H. 古可魚語 (cont.)

- 如果好好寫，行數會在 150 ∼ 200 左右。
- 不過建議加上各種例外處理，方便 debug ，所以會稍長一點。
- 輸出範例是不會過的

題目 H. 古可魚語 (cont.)

- 這個語言其實非常強
- 例如，測資中有國中組 pE 的 solution.

題目 H. 古可魚語 (cont.)

- 你知道嗎？如果讓古可魚語支援 long long ...
- Solution of pA:

```
(define == (lambda (a b) (if (< a b) 0 (if (< b a) 0 1))))

(define min (lambda (a b) (if (< a b) a b)))
(define max (lambda (a b) (if (< a b) b a)))

(define pair (lambda (a b) (lambda (m) (m a b))))
(define first (lambda (p) (p (lambda (a b) a))))
(define second (lambda (p) (p (lambda (a b) b))))

(define [] (pair 1 0))
(define : (lambda (x xs) (pair 0 (pair x xs))))
(define empty? (lambda (xs) (first xs)))
(define head (lambda (xs) (first (second xs))))
(define tail (lambda (xs) (second (second xs))))
(define !!
  (lambda (xs idx)
    (if (< 0 idx)
      (!! (tail xs) (- idx 1))
      (head xs))))
```

題目 H. 古可魚語 (cont.)

```
; fast mul
(define *
  (begin
    (define _*
      (lambda (a b c)
        (if (< b c)
          (pair b 0)
          (begin
            (define tmp (_* (+ a a) b (+ c c)))
            (if (< (first tmp) c)
              tmp
              (pair (- (first tmp) c) (+ (second tmp) a)))))))
    (lambda (a b)
      (second (_* a b 1)))))

(define mklist
  (lambda (len init)
    (if (< 0 len)
      (: init (mklist (- len 1) init))
      [])))
(define set
  (lambda (xs idx val)
    (if (< 0 idx)
      (: (head xs) (set (tail xs) (- idx 1) val))
      (: val (tail xs)))))
```

題目 H. 古可魚語 (cont.)

```
(define INF 5145145145145145145)
(define dijkstra
  (lambda (n s t edge)
    (begin
      (define fmin
        (lambda (idx xs cidx cval)
          (if (empty? xs)
            cidx
            (if (< (head xs) cval)
              (fmin (+ idx 1) (tail xs) idx (head xs))
              (fmin (+ idx 1) (tail xs) cidx cval)))))
      (define _dijkstra
        (lambda (dist done)
          (begin
            (define cur (fmin 1 (tail dist) 0 (head dist)))
            (define done (set done cur 1))
            (define update
              (lambda (dist edge _dist)
                (if (empty? edge)
                  dist
                  (begin
                    (define ele (head edge))
                    (define des (first ele))
                    (define ndist (+ _dist (second ele)))
```

題目 H. 古可魚語 (cont.)

```
                    (if (!! done des)
                      (update dist (tail edge) _dist)
                      (if (< ndist (!! dist des))
                        (update (set dist des ndist) (tail edge) _dist)
                        (update dist (tail edge) _dist))))))
              (if (== cur t)
                (!! dist cur)
                (_dijkstra (set (update dist (!! edge cur) (!! dist cur)) cur INF) done)
                    ))))
        (_dijkstra (set (mklist n INF) s 0) (mklist n 0)))))

(define senior-a-sol
  (lambda (n s t f edges)
    (begin
      (define cost
        (lambda (c d nc)
          (+ (* c (min d f)) (* nc (max 0 (- f d))))))
      (define edge-maker
        (lambda (edge edges)
          (if (empty? edges)
            edge
            (begin
              (define use (head edges))
              (define _s (- (!! use 0) 1))
              (define _e (- (!! use 1) 1))
```

題目 H. 古可魚語 (cont.)

```
            (define _cst (cost (!! use 2) (!! use 3) (!! use 4)))
            (edge-maker (set edge _s (: (pair _e _cst) (!! edge _s))) (tail edges)))
                ))
      (dijkstra n (- s 1) (- t 1) (edge-maker (mklist n []) edges)))))

; sample data
(define edges (: (: 1 (: 2 (: 1 (: 1 (: 1 [])))))(: (: 2 (: 4 (: 5 (: 1 (: 3 [])))))(:
      (: 1 (: 3 (: 1 (: 1 (: 1 [])))))(: (: 3 (: 4 (: 6 (: 1 (: 1 [])))))[])))))
(display (senior-a-sol 4 1 4 1 edges))
(define edges (: (: 1 (: 2 (: 1 (: 1 (: 1 [])))))(: (: 2 (: 4 (: 5 (: 1 (: 3 [])))))(:
      (: 1 (: 3 (: 1 (: 1 (: 1 [])))))(: (: 3 (: 4 (: 6 (: 1 (: 1 [])))))[])))))
(display (senior-a-sol 4 1 4 2 edges))
(define edges (: (: 1 (: 2 (: 5 (: 1 (: 4 [])))))[]))
(display (senior-a-sol 2 1 2 2 edges))
```

Thank You!