UVa 11516 WiFi

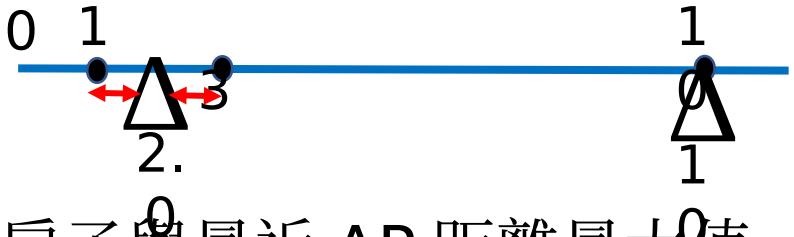
UVa 11516 WiFi (Time Limit: 1 second)

無線網路 WiFi: 大街上的居民開會決定要在他們居住的街上安裝無線網路，讓無線網路環境涵蓋所有住戶，請你幫忙選擇無線網路基地台 (AP) 的地點，他們希望訊號愈強愈好，但他們購買 AP 的預算有限，在有限的 AP 數量之下，使得在「所有房子與其最近的 AP 之間的距離」中最大值，愈小愈好。

大街是直線的，每間房子的門牌號碼剛好等於與端點的距離，例如 123 號的住戶，距離大街的起點為 123 公尺。

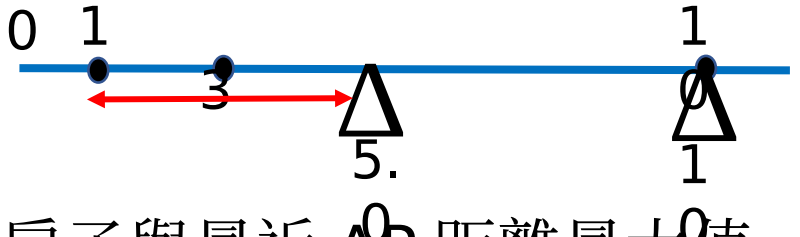
Test Case #1

3 間房子與 2 個 AP



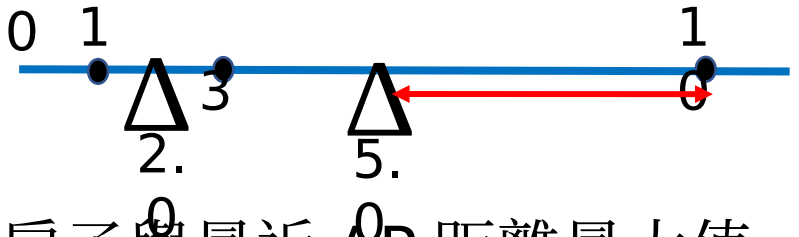
房子與最近 AP 距離最大值：1.0 √

最大距離最小化



房子與最近 AP 距離最大值：4.0

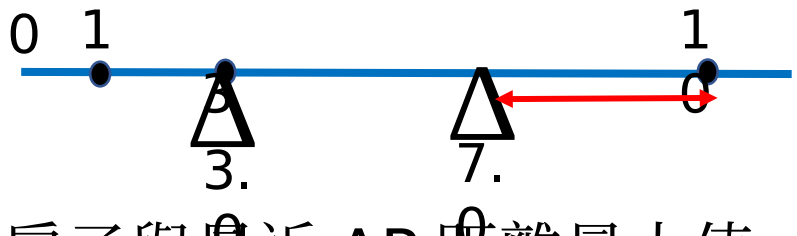
房子與最近 AP 距離最大值：2.0



房子與最近 AP 距離最大值：5.0

- 房子所在位置

Δ AP 所在位置



房子與最近 AP 距離最大值：3.0

Sample Input

```
1
2 3
1
3
10
```

Test Case 數目

AP 數目 , 房子數目

Test Case

房子位置 (或座標)

Sample Output

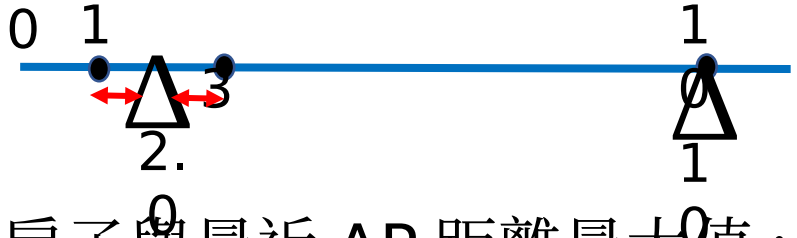
```
1.0
```

所有房子與其最近的 AP 之間的距離中最大值最小化，輸出此值 (顯示到小數點以下一位數)

AP: access point

Test Case #1

3 間房子與 2 個 AP



房子與最近 AP 距離最大值 : 1.0

- 房子所在位置

Δ AP 所在位置

Solution

- 從許多的最大值中找最小的問題
- Bisection Method (二分法)
- 由於輸出距離要到小數點以下一位 , 要進行 2 次二分法找答案
  - 整數運算
- 為配合二分法需有一 Greedy Method 做狀況判斷

觀察

- 如果已有的 Ap 數目≧房子數目，則在每間房子位置放一個 Ap, 如此一來離房子最近 Ap 的距離都是 0. 因此，本題答案輸出 0.0.
- 不然（已有的 Ap 數目 < 房子數目）用二分法找答案 .

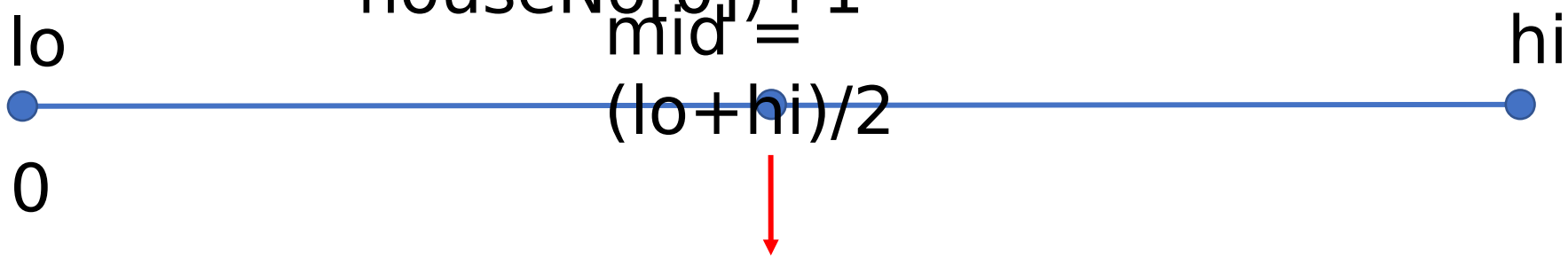
Bisection Method (二分法)

Bisection Method

Initiall lo = 0 hi = 2*(HouseNo[house-1]-houseNo[0])+1

hi= 頭尾房子距離的 2 倍

Step 1



Greedy check(2*mid)

= True: mid 值要減少（需要的 AP 數目）

= False: mid 值要增長（需要的 AP 數目）

- 給定檢驗 wifi 範圍，看看需要的 AP 數目是否會超過已有的 AP 數目？

2*mid : wifi 範圍 mid: 房子至最近 AP 最大距離

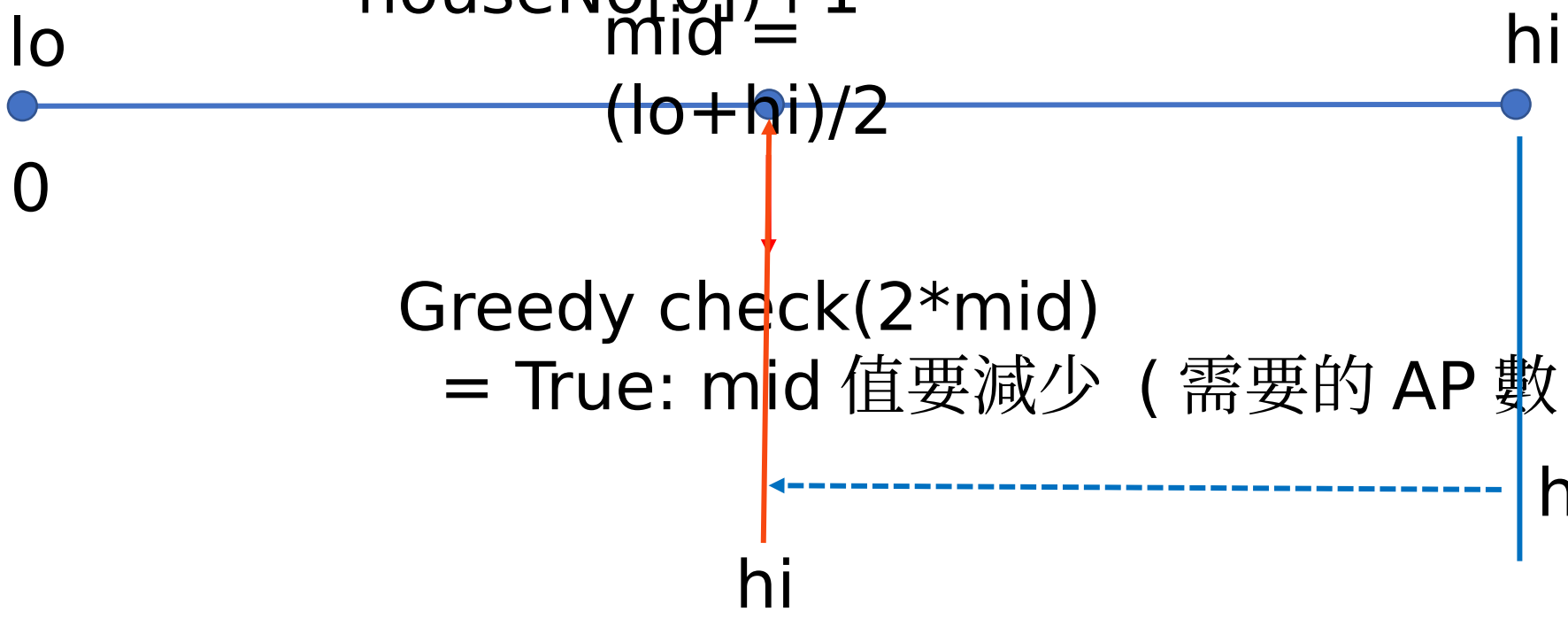
Bisection Method

Initiall lo = 0

hi = 2*(HouseNo[house-1]-houseNo[0])+1

hi= 頭尾房子距離的 2 倍

Step 2



2*mid : wifi 範圍

mid: 房子至最近 AP 最大距離

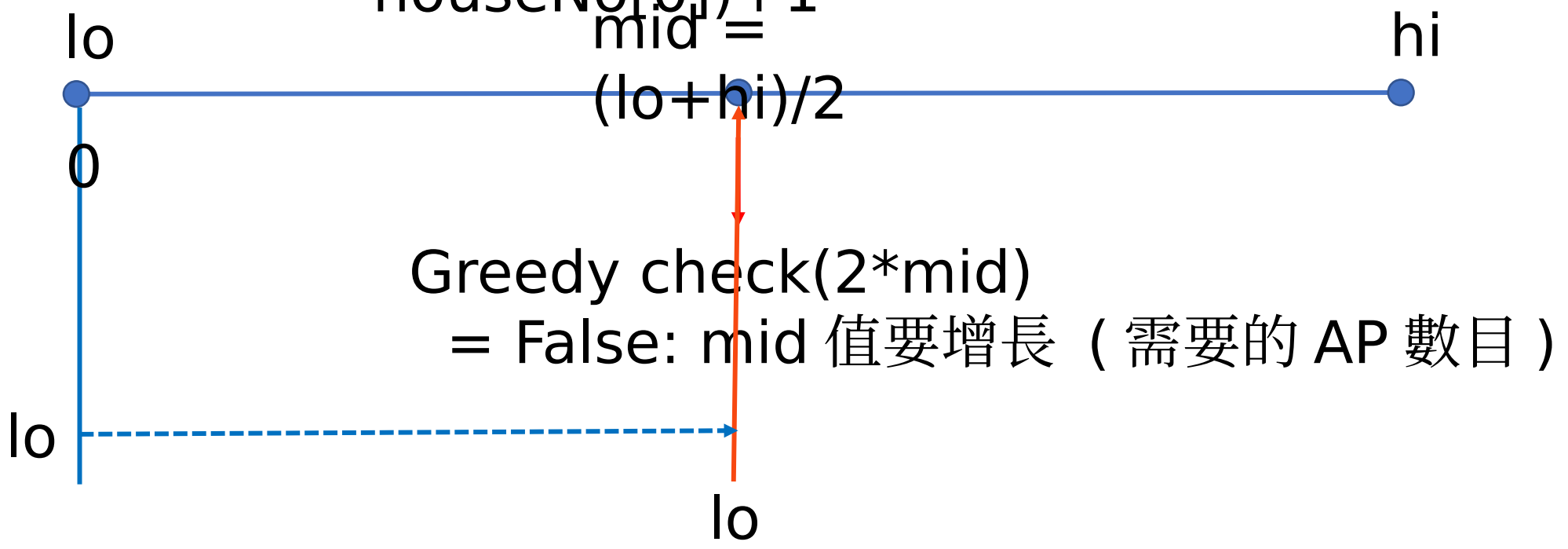
Bisection Method

Initiall lo = 0

hi = 2*(HouseNo[house-1]-houseNo[0])+1

hi= 頭尾房子距離的 2 倍

Step 2’



2*mid : wifi 範圍 mid: 房子至最近 AP 最大距離

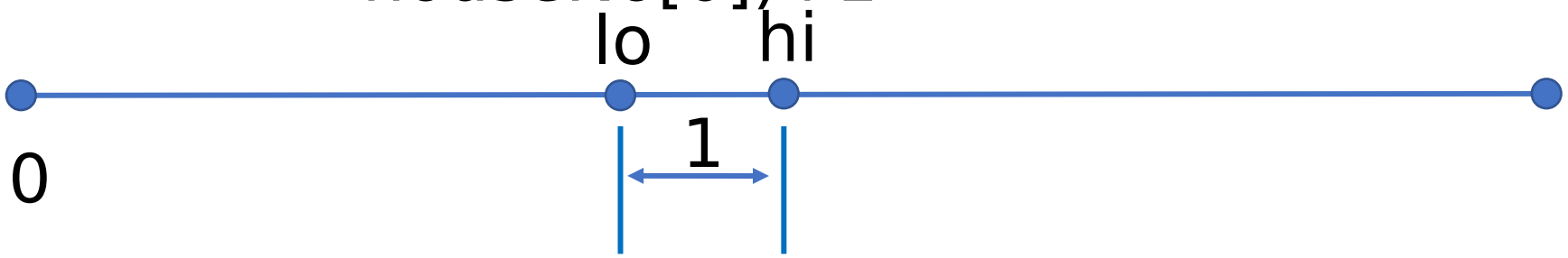
Bisection Method

Initiall lo = 0 hi = 2*(HouseNo[house-1]-houseNo[0])+1

Step 1

XX

hi= 頭尾房子距離的 2 倍



由於輸出距離要到小數點以下一位，所以當 hi-lo=1 時，要再進行第 2 次二分法找距離精準到小數點以下一位。

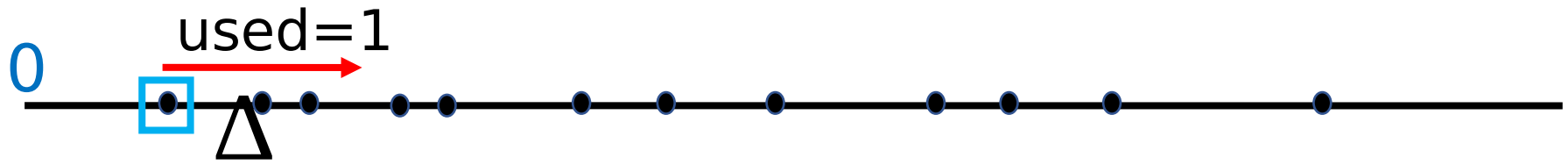
- 為了與第一次二分法一樣採整數運算
- 把所有的數據乘以 10 (包含此時的 lo 與 hi 值 , 房子位置) 進行第二次二分法，直到 hi-lo=1 時，輸出距離的整數是 hi/10, 小數部分是 hi %10。
  - 為何最後的距離答案採用 hi 值取得 ?
  - 因為二分法過程中每次調整到 hi 值時 , 包含一個狀況：根據給定的 wifi 範圍算出需要的 AP 個數與已有的 AP 個數相同，而 lo 值調整就無此特性。

Greedy Check

- 給定 wifi 範圍，看看需要的 AP 數目是否會超過已有的 AP 數目？

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

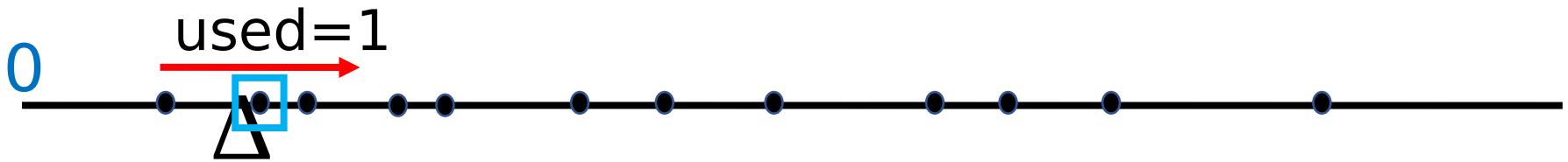
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used: 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

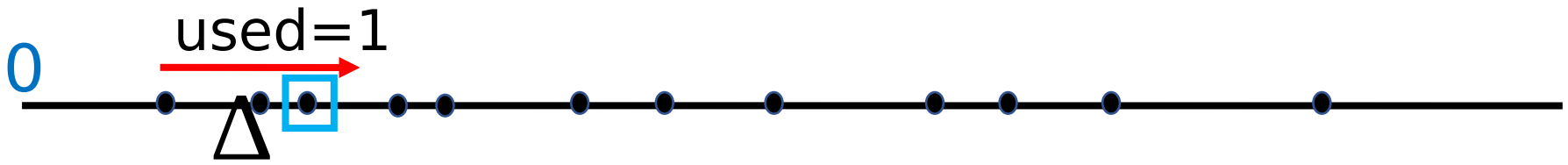
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used: 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

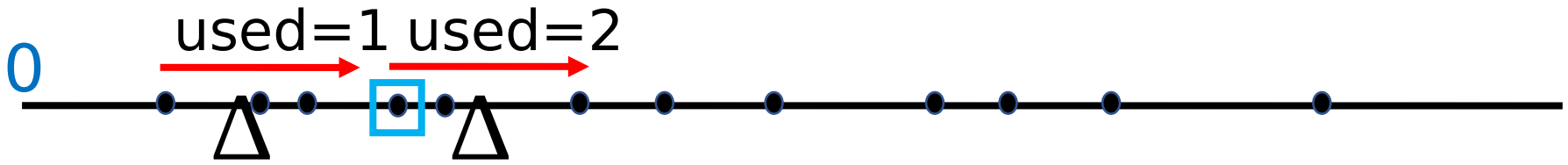
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used: 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

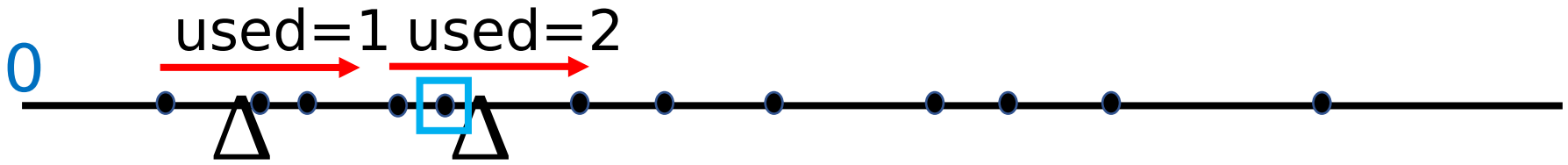
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

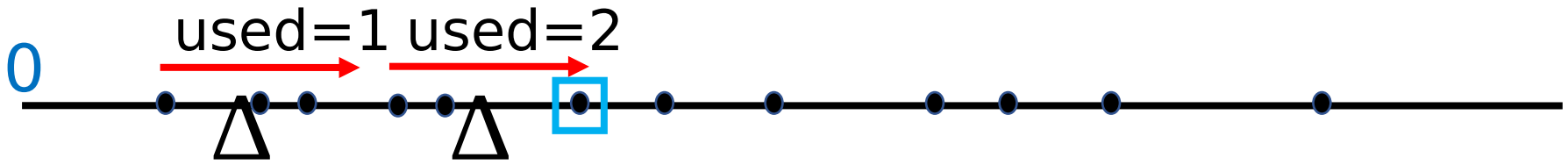
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

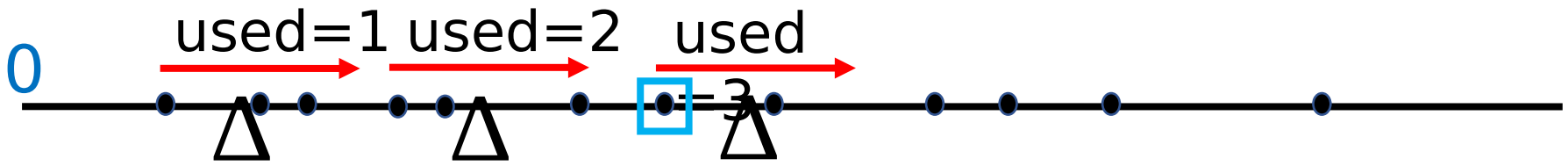
- 房子所在位置

Δ AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

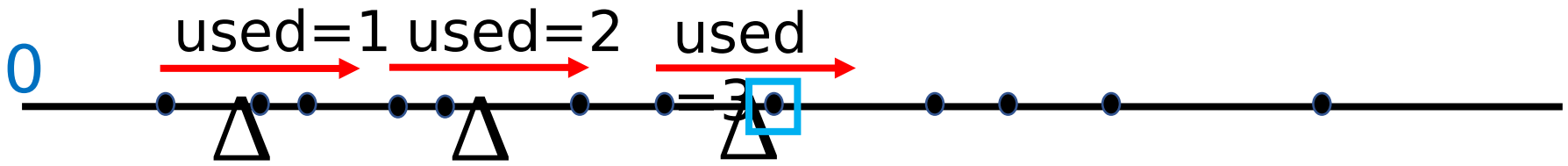
房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

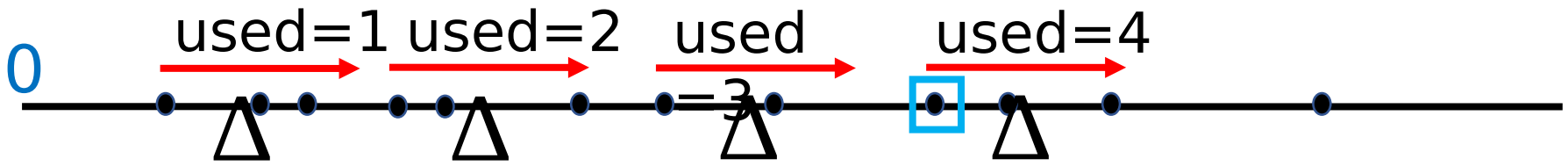
房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

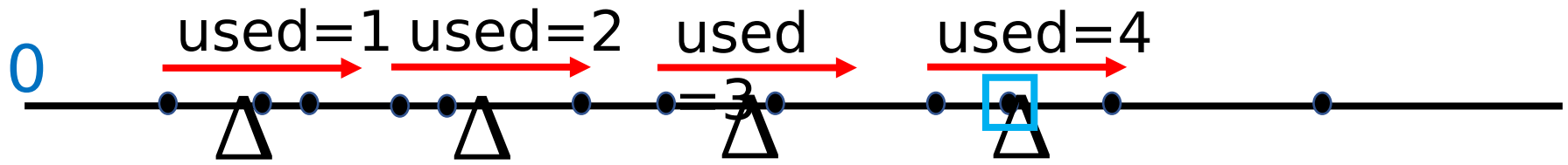
房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

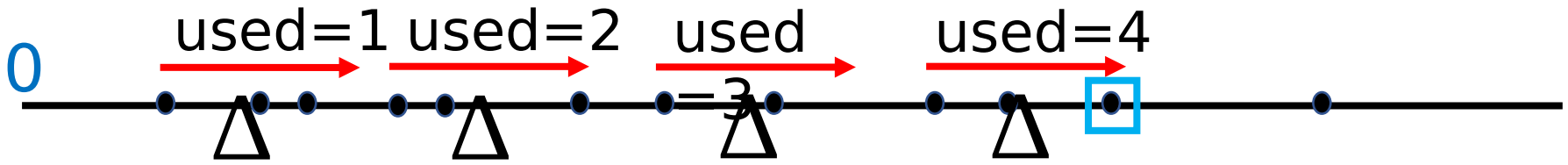
房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

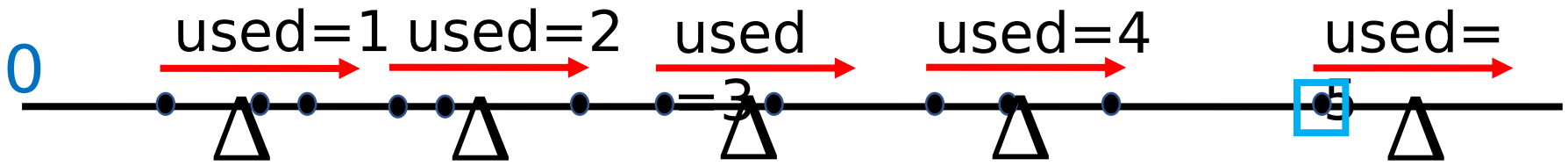
房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Greedy Check

給定檢驗長度 (wifi 範圍), 看看需要的 AP 數目是否會超過已有的 AP 數目 ?



最後 , 如果 used , 則要再檢驗短一點的長度 ;

不然要再檢驗長一點的長度 (used ;

檢驗長度 (房子至最近 AP 最大距離)

wifi 範圍 (是檢驗長度的 2 倍)

房子所在位置

AP 所在位置 (放在 wifi 範圍中點)

used : 需要的 AP 數目

Uva 11516 Code (1/3)

```
#include <cstdio>
#include <algorithm>
using namespace std;

int Ap, house;                          // Ap: 已知的 Ap 數目 , house: 房子數目
int houseNo[100000+10];    // houseNo[i]: 第 i 間房子的位置

bool greedycheck(int mid)   //  檢視 wifi 範圍 mid 所需 AP 數目是否 <= 已知 AP 數目
{
    int used = 1;                                   // used: 需要的 AP 數目
    int wifirange = houseNo[0] + mid;   //  第一個 Ap 放在 houseNo[0]+mid/2 位置
    for ( int i = 0; i < house; i++ ) {
        if ( houseNo[i] > wifirange ) { wifirange = houseNo[i] + mid; used++; }
                                                    //  第 i 間房子超出 wifi 範圍 , 要新增一個 Ap
    }
                                                    // 第 used 個 Ap 放在 houseNo[i]+mid/2 位置
```

Uva 11516 Code (2/3)

```
int main ()
{
    int testCase;
    freopen("11516.in","r",stdin);  freopen("11516.out","w",stdout);
    scanf ("%d", &testCase);
    while ( testCase-- ) {
        scanf ("%d %d", &Ap, &house);
        for ( int i = 0; i < house; i++ ) scanf ("%d", &houseNo[i]);
        if ( Ap >= house ) { printf ("0.0\n"); continue; }  // 已有的 Ap 數目 >= 房子數目
                                                            第一次二分法
        sort (houseNo, houseNo + house);   //  位置由小至大排列
        int lo = 0;
        int hi = 2*(houseNo[house-1]-houseNo[0])+1;   // hi= 頭尾房子距離的 2 倍
        while ( hi - lo > 1 ) {
            int mid = (lo + hi) / 2;
            if ( greedycheck(mid * 2) ) hi = mid;   // wifi 範圍 mid*2 所需 AP 數目 <= 已知 AP 數目 ,
                                //  縮短 wifi 範圍測試
            else lo = mid;    // 增長 wifi 範圍測試
```

Uva 11516 Code (3/3)

第二次二分法

```
    lo *= 10;     //  相關資料放大 10 倍： 為求數值精度到小數點以下一位
    hi *= 10;     //  維持整數運算
    for ( int i = 0; i < house; i++ ) houseNo[i] *= 10;
    while ( hi - lo > 1 ) {
        int mid = (lo + hi) / 2;
        if ( greedycheck(mid * 2) ) hi = mid; // wifi 範圍 mid*2 所需 AP 數目 <= 已知 AP 數目 ,
                        //  縮短 wifi 範圍測試
        else lo = mid;    // 增長 wifi 範圍測試
    }
    printf ("%d.%d\n", hi / 10, hi % 10);   //  輸出整數與小數 ( 一位 ) 部分
  }
  return 0;
}
```