LA 4445 A Careful Approach

(ACM ICPC World Finals Stockholm 2009, UVa 1079)

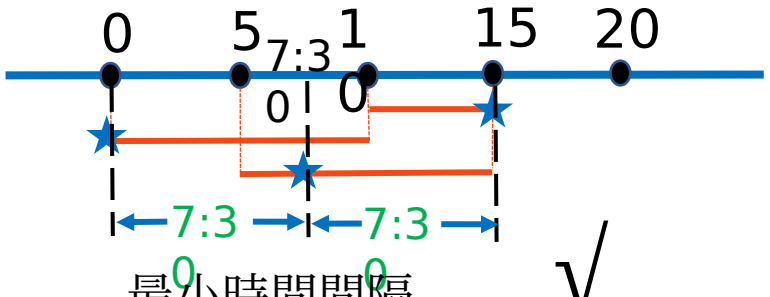
LA 4445 A Careful Approach (Time Limit: 4.5 seconds)

機場飛機著陸安排：機場有 n (2 ≤ n ≤ 8) 架飛機準備降落，根據每架飛機各種條件，它們各有安全著陸的時間區間 [ai,bi] (0 ≤ ai ≤ bi ≤ 1440, 時間單位：分)，請安排所有飛機的最佳安全降落時間使得飛機最短的著陸時間間隔最大化，輸出最大的最小間隔著陸時間。

Test Case #1

飛機安全著陸時間區間
(單位 : 分)

```
0 10
5 15
10
15
```

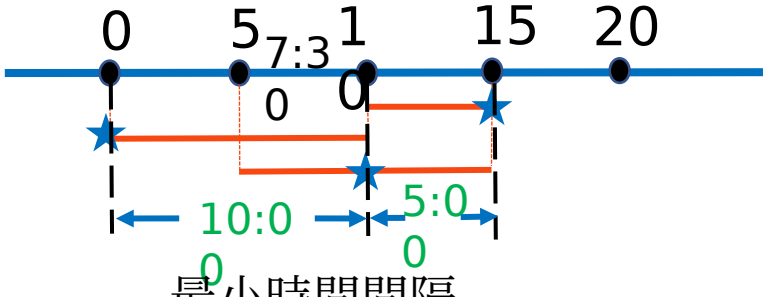


最小時間間隔

7:30

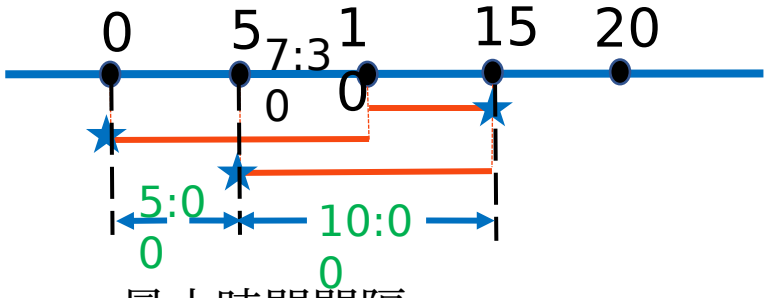
最小時間間隔最大值

√



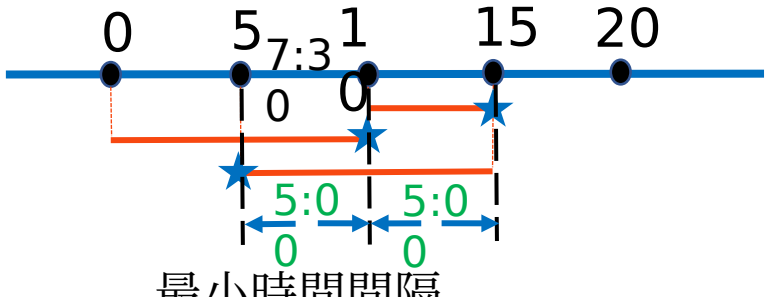
最小時間間隔

5:00



最小時間間隔

5:00



最小時間間隔

5:00

★: 著陸時間點

Sample Input

```
3
0 10
5 15
10
15
2
0 10
10
20
0
```

飛機數量

Test Case #1

飛機安全著陸時間區間
(單位：分)

Test Case #2

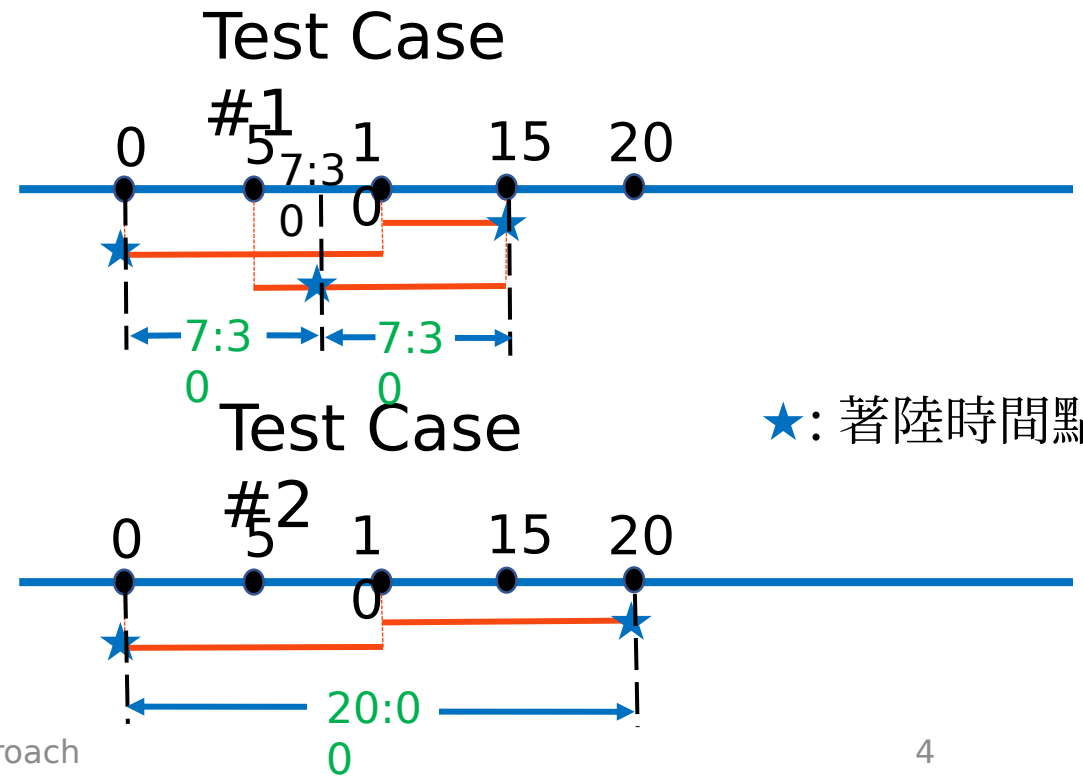
Test Case 輸入結束

Sample Output

```
Case 1: 7:30
Case 2: 20:00
```

(分：秒)

飛機著陸時間最小間隔最大化之間隔時間



Solution

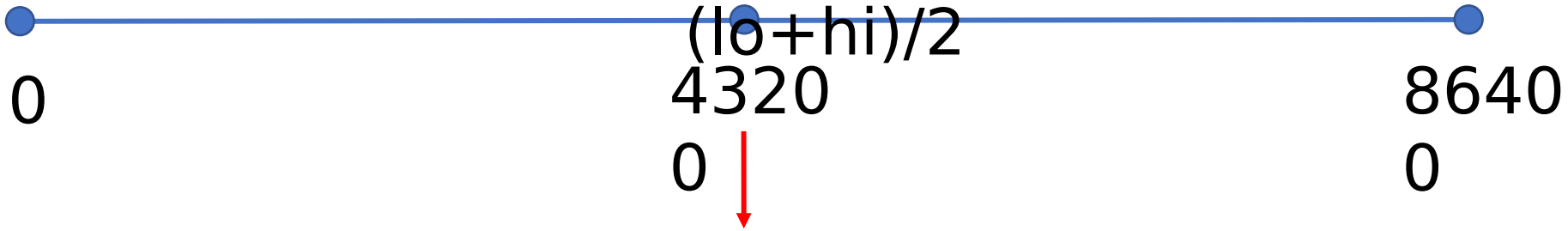
- Complete Search
  - 因為 n (2 ≤ n ≤ 8) 架飛機 , n 不大 , 直接把所有可能的著陸順序 (最多 8!=40320) 每一順序的最小間隔時間找出來 , 然後最大的最小間隔時間就得到了。
- Bisection method (二分法)
  - 給定一個飛機著陸順序 , 利用二分法左右逼近方式將最小間隔時間找出來。
- Greedy Landing (Greedy Method)
  - 給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短 ?

Bisection Method (二分法)

Bisection Method

Initiall lo = 0 hi = 86400 (1440*60 秒 / 日)

Step 1



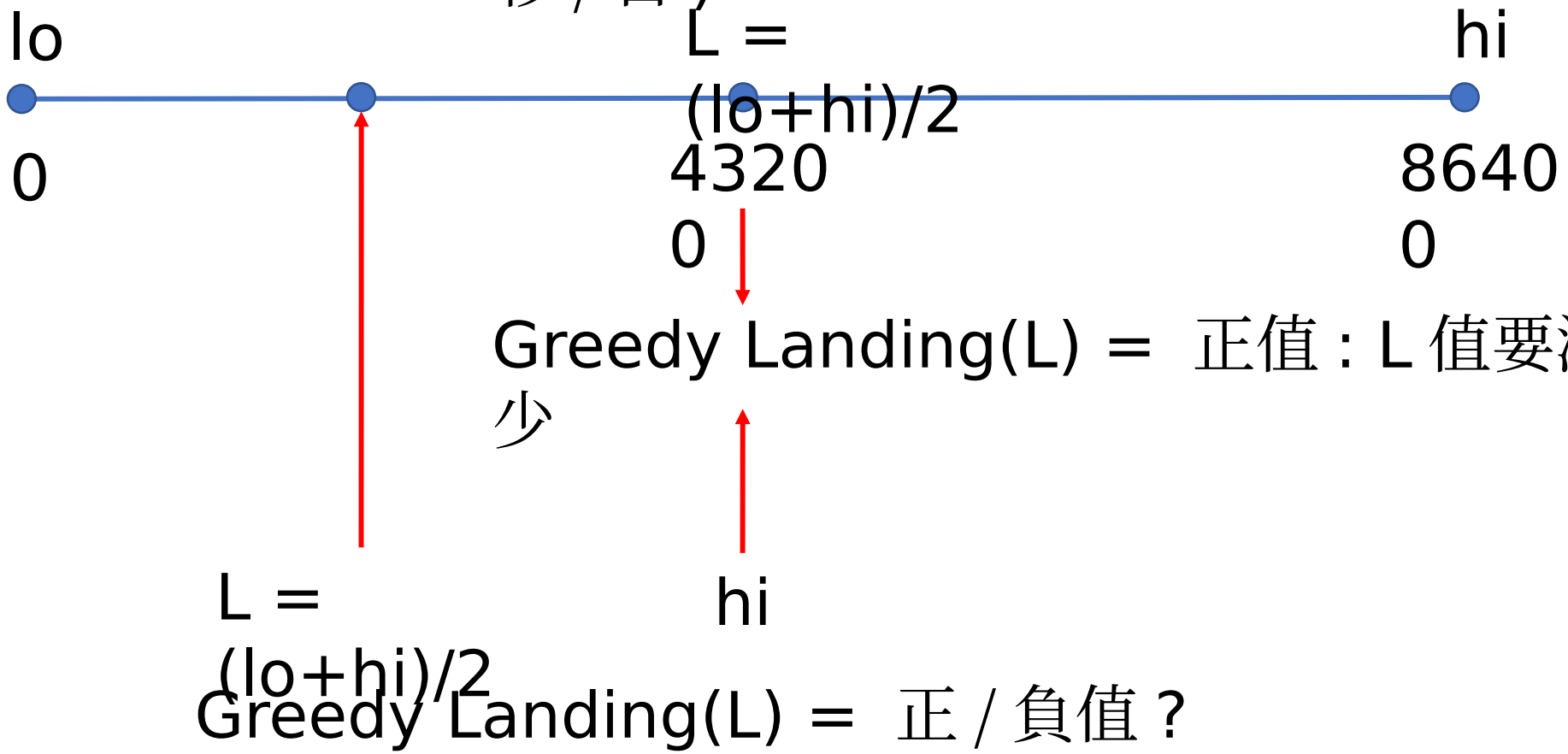
Greedy Landing(L) = 正值 : L 值要減少

= 負值 : L 值要增長

Bisection Method

Initiall lo = 0 hi = 86400 (1440*60 秒 / 日)

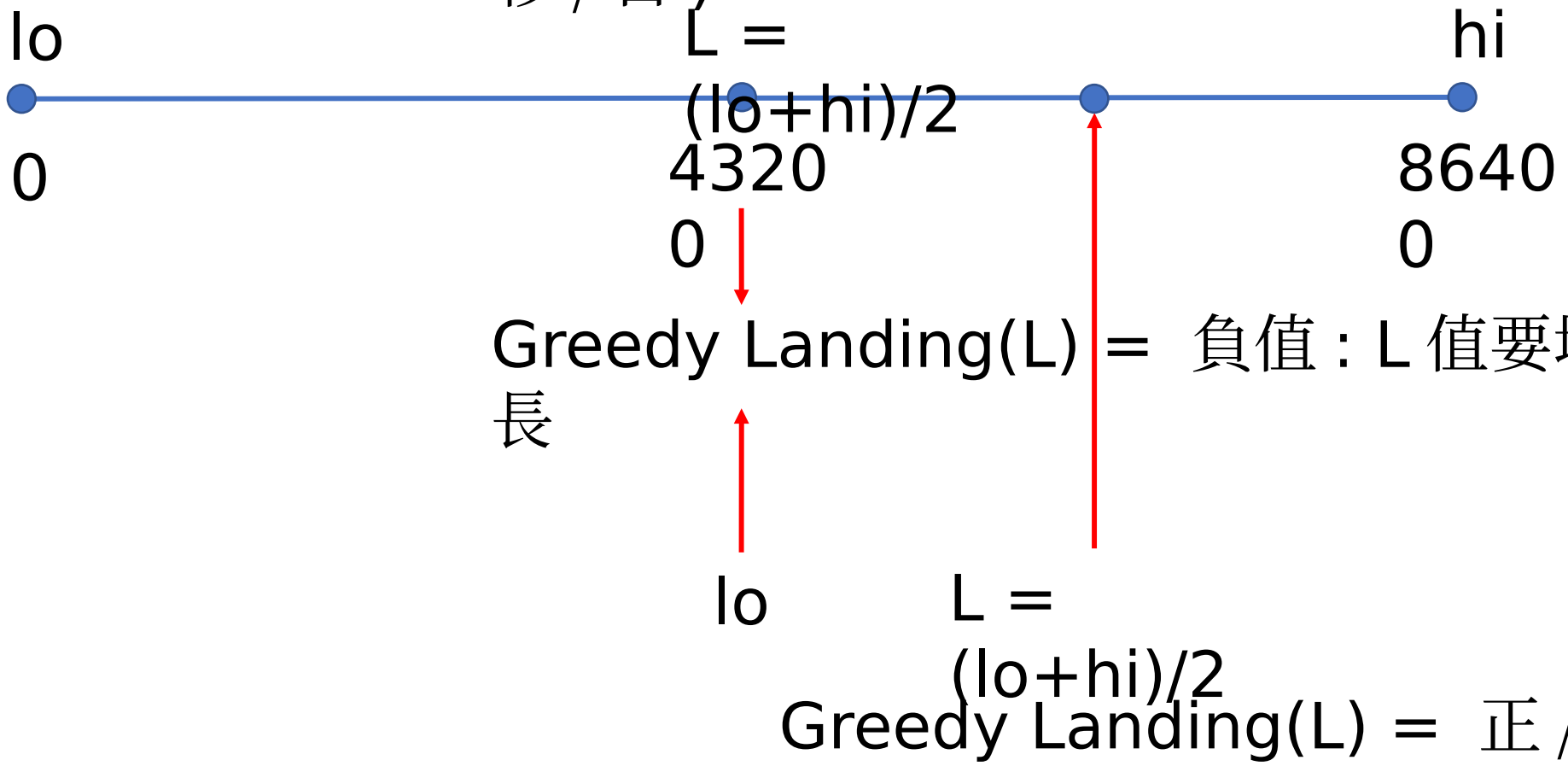
Step 2



Bisection Method

Initiall lo = 0 hi = 86400 (1440*60 秒 / 日)

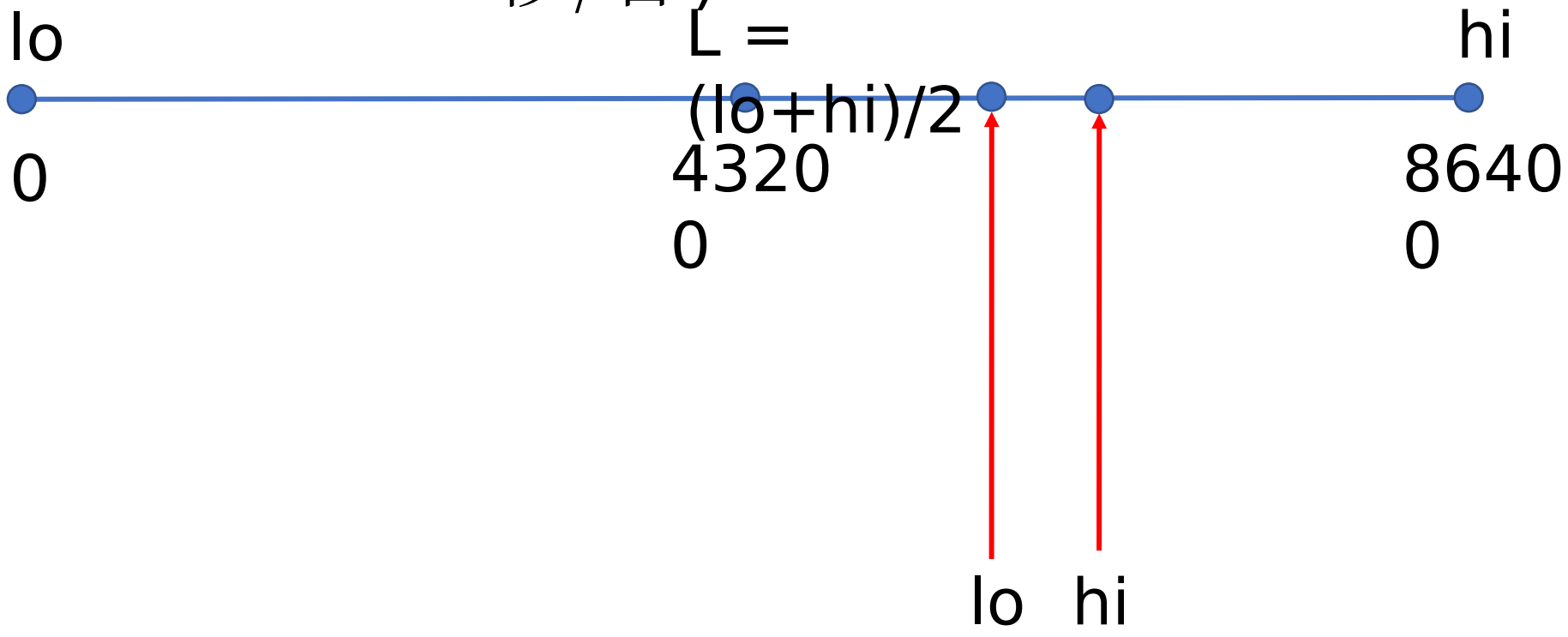
Step 2’



Bisection Method

Initiall lo = 0 hi = 86400 (1440*60 秒 / 日)

Step xx



|lo-hi| < 就停止二分法逼近

認為找到的答案 (最小的間隔時間) 是 (lo+hi)/2

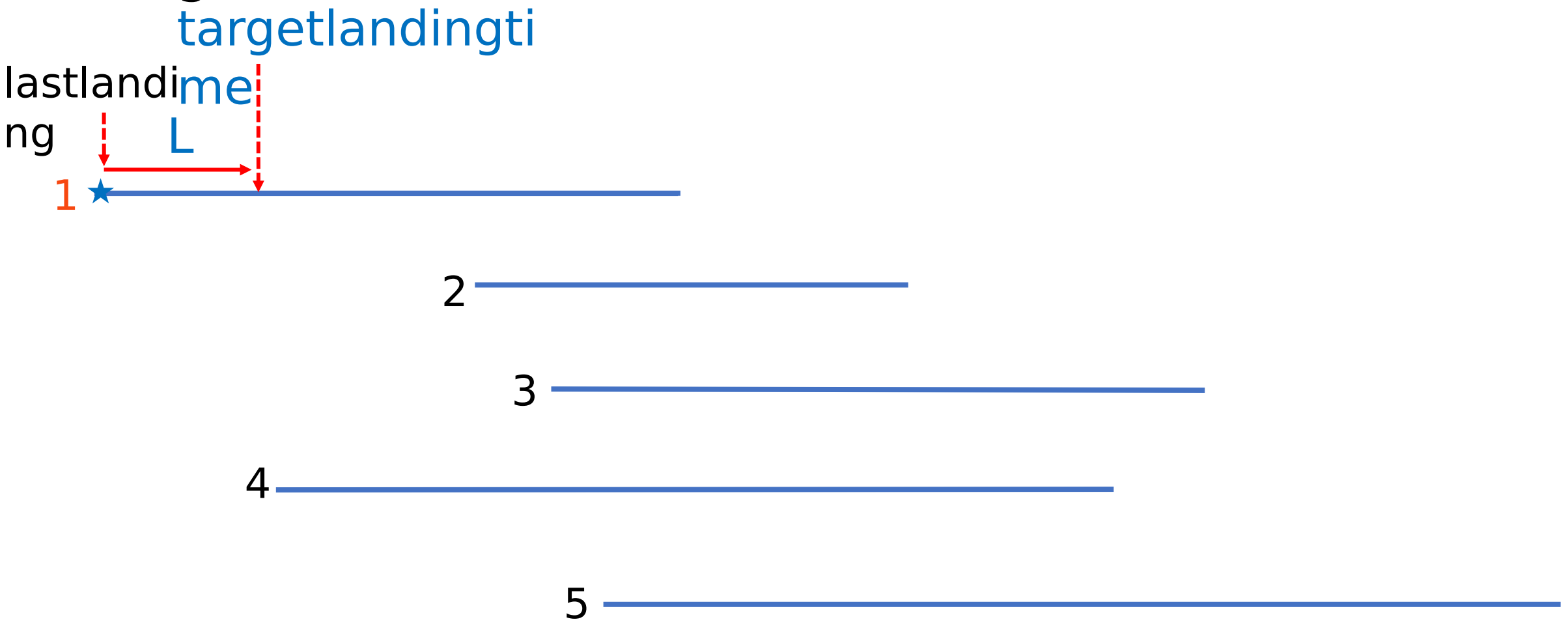
Greedy Landing

給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？

- Case 1 : L 還要增長

Greedy Landing

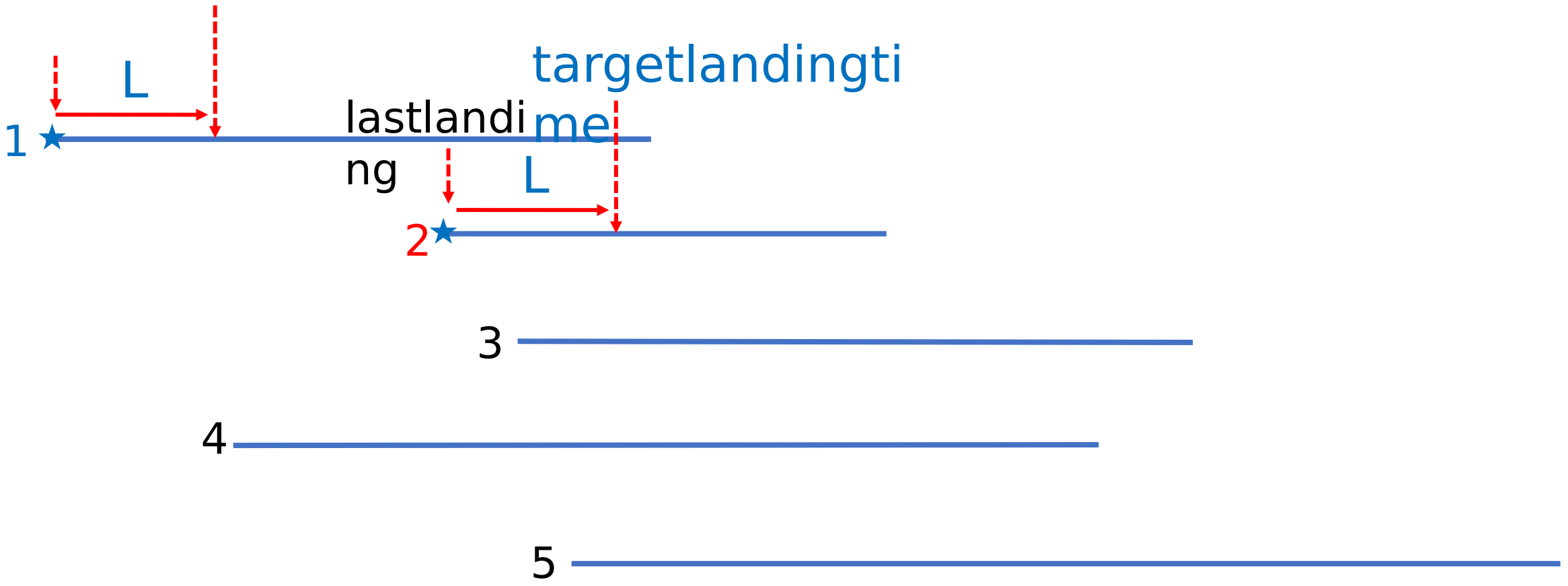
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

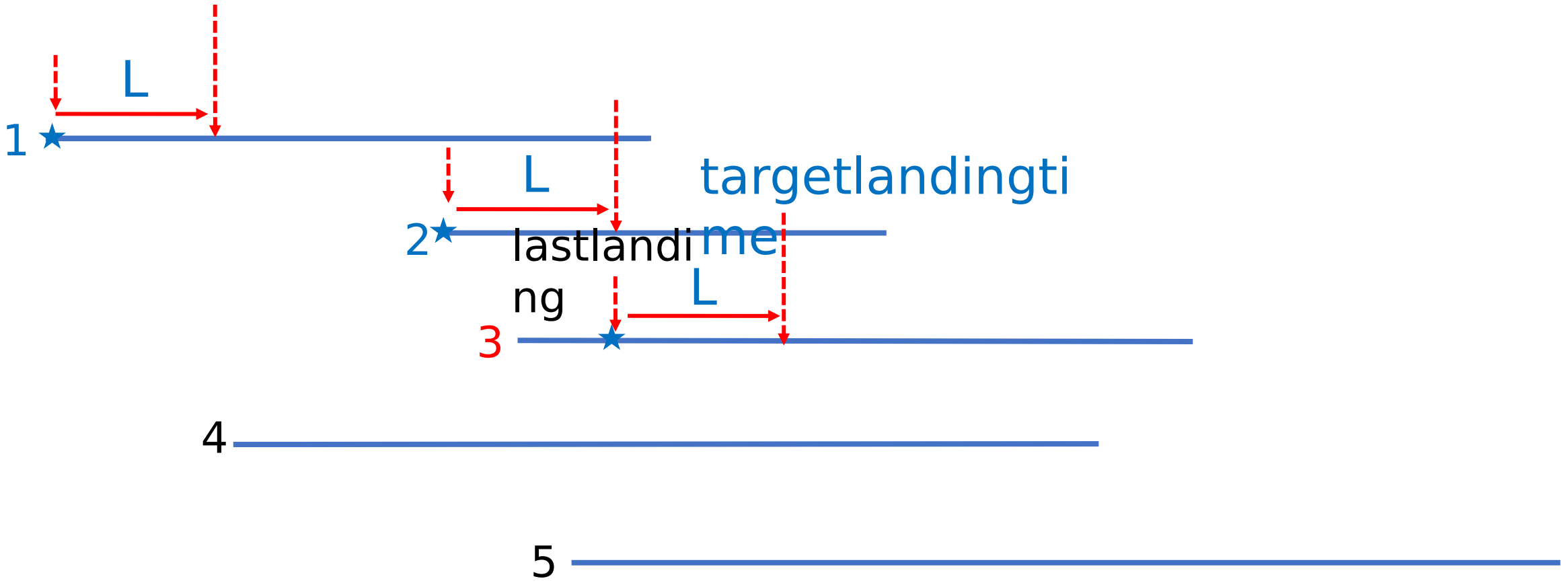
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

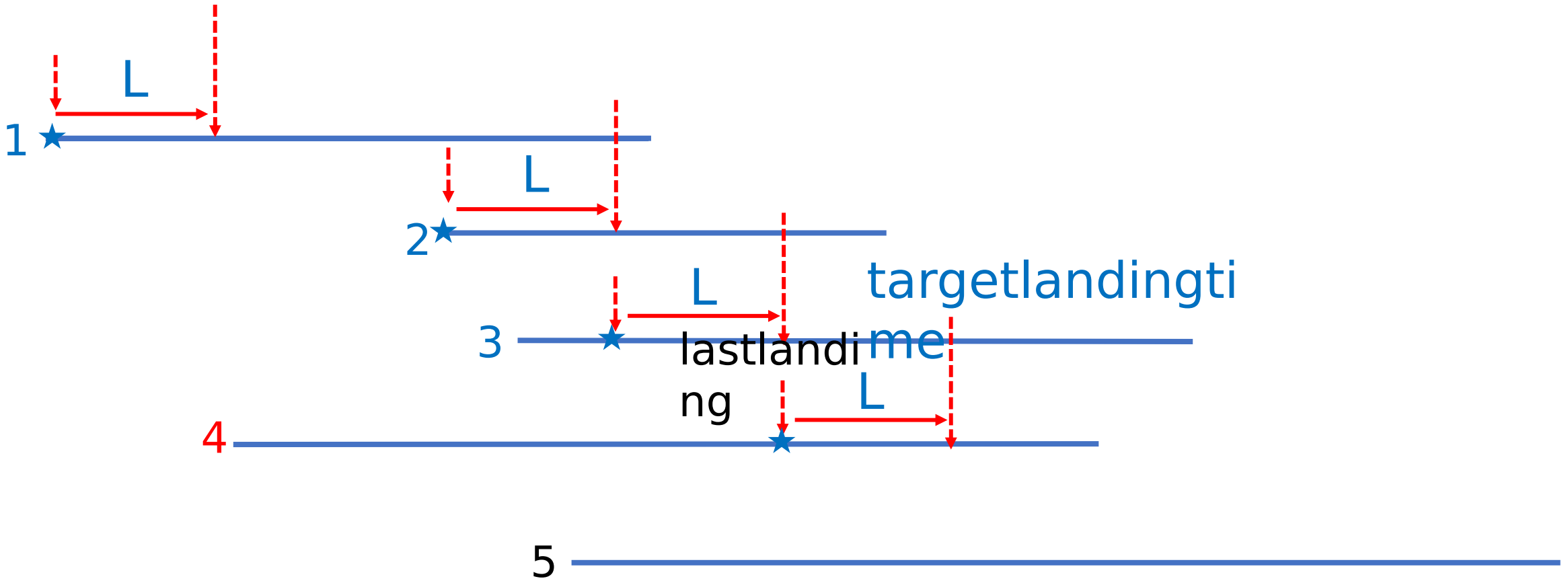
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

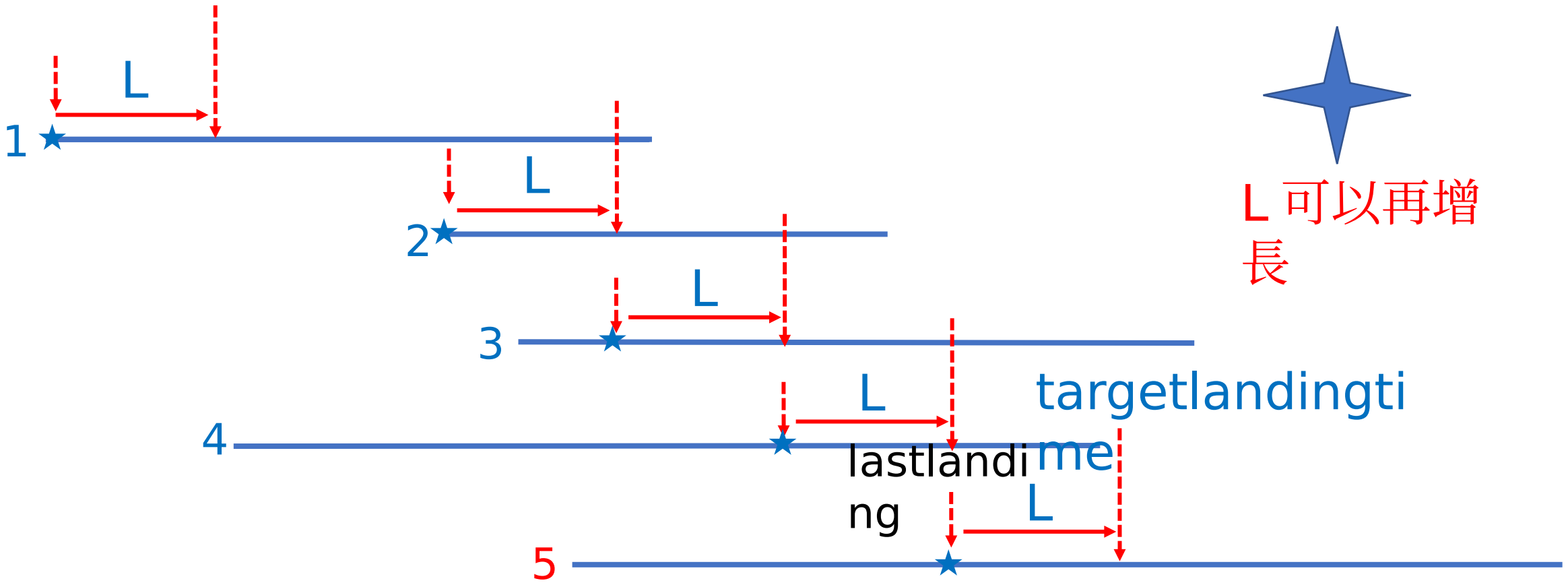
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

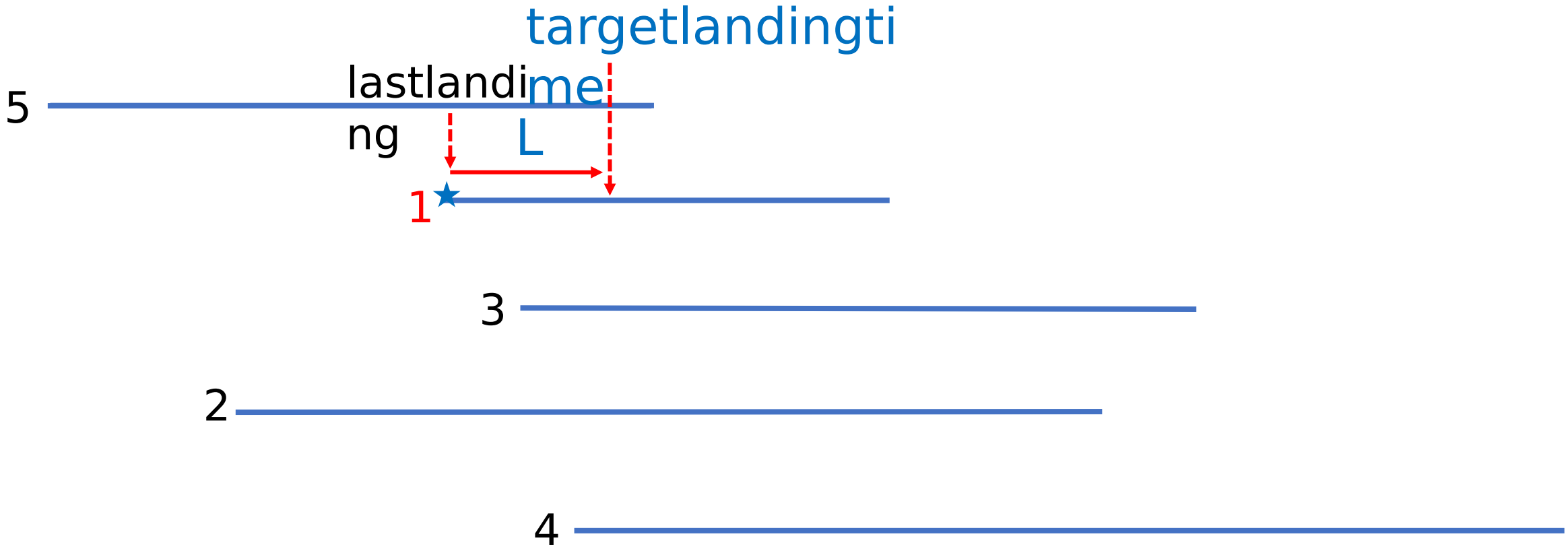
Greedy Landing

給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？

- Case 2 : L 還要減短

Greedy Landing

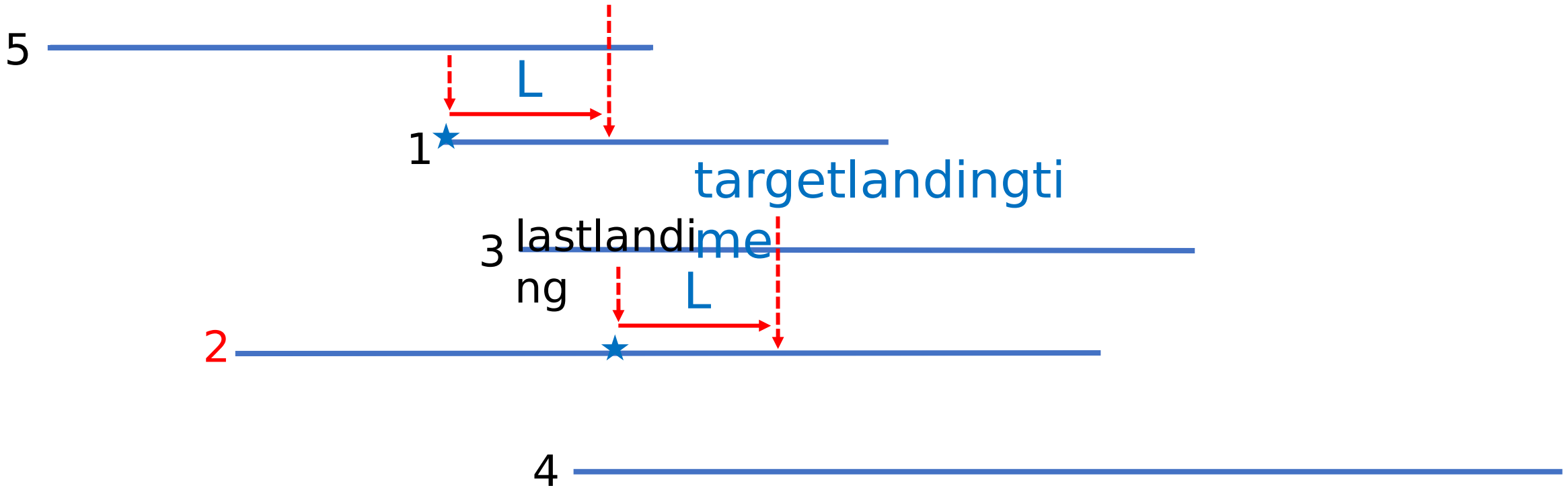
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

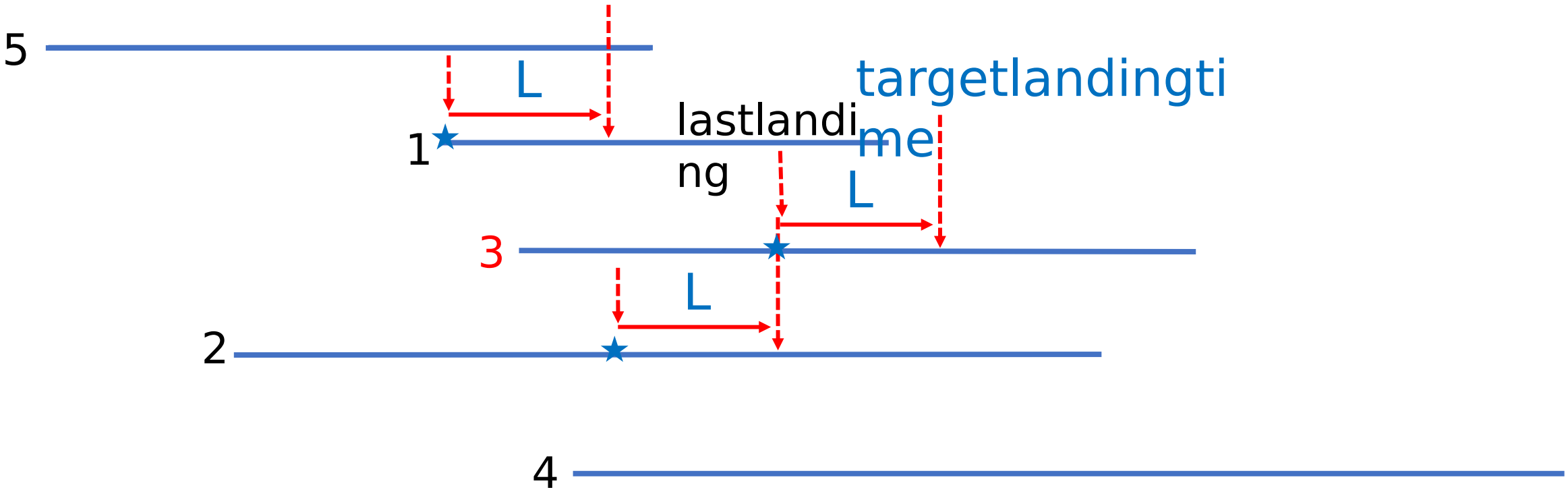
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

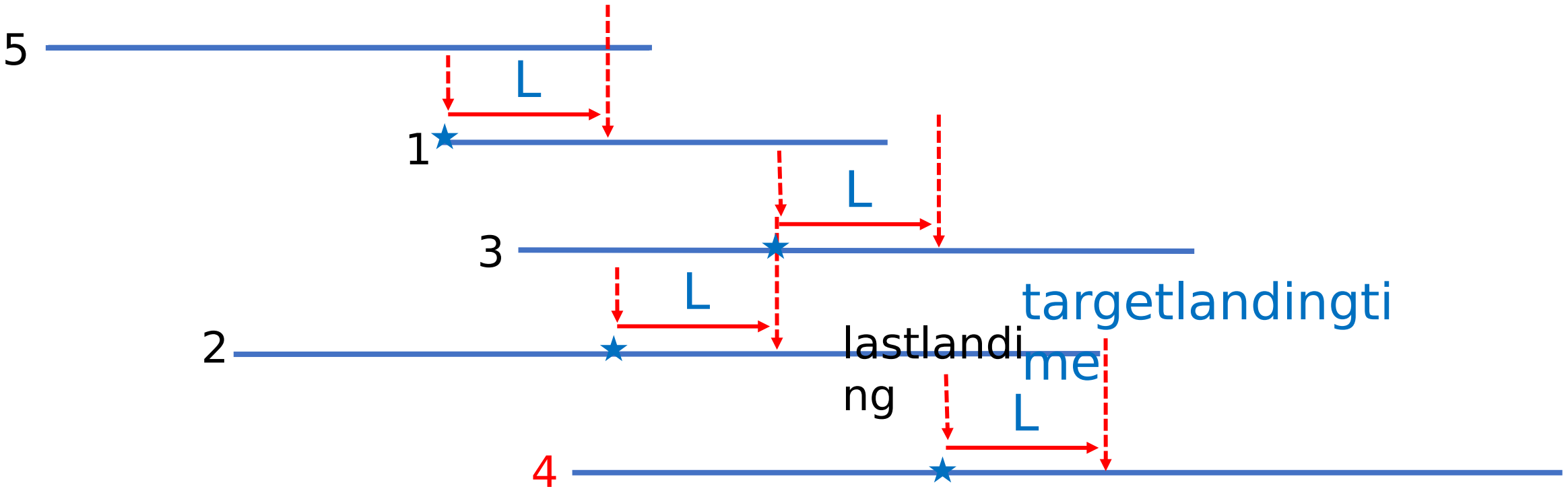
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

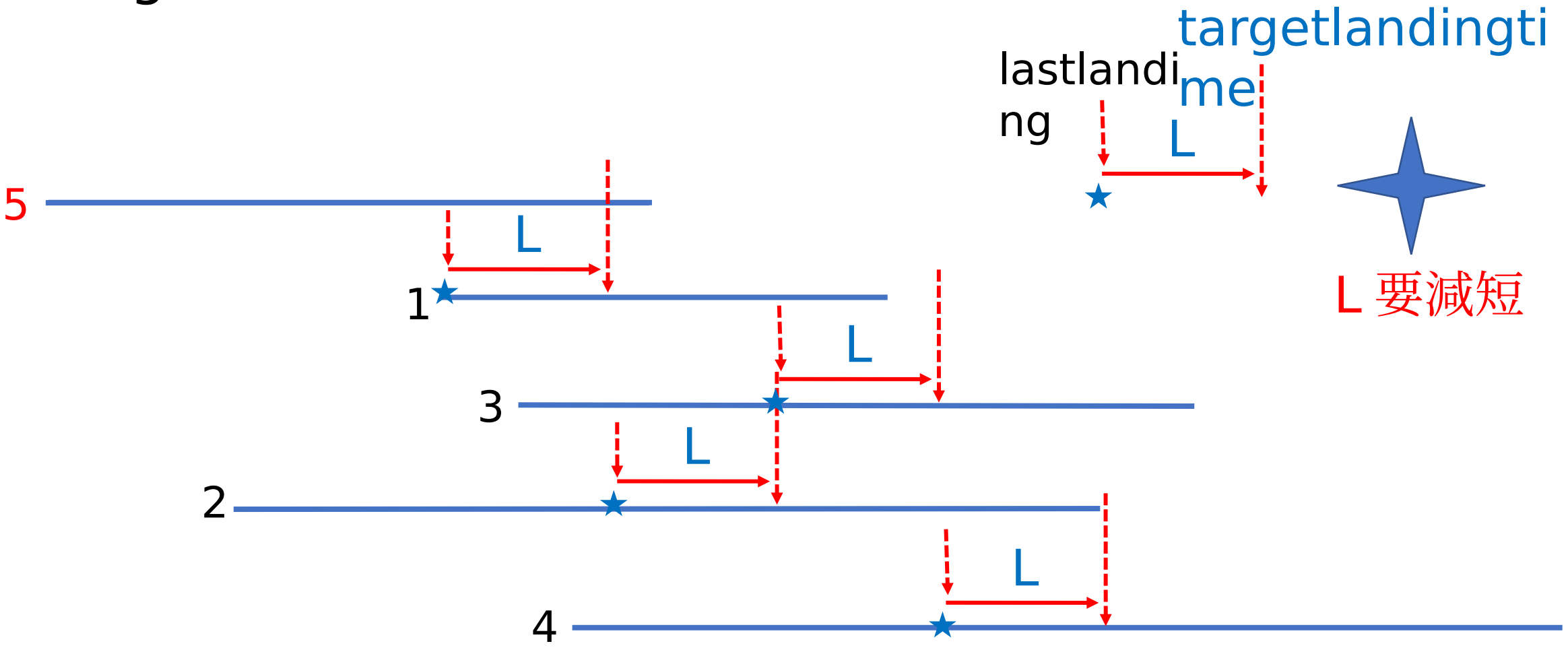
給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

Greedy Landing

給定一個飛機著陸順序與時間間隔 L, 看看時間間隔 L 是可以再增長還是減短？



★: 著陸時間點

LA 4445 Code (1/3)

```
#include <algorithm>
#include <cmath>
#include <cstdio>
using namespace std;
int i, n, caseNo = 1, order[8];     // order[8]: 存飛機著陸順序
double a[8], b[8], L, maxL;         // (a[i],b[i]): 順序 i 飛機的安全降落時間區間
double greedyLanding() {  // 給定一個飛機著陸順序與時間間隔 L,
                // 看看時間間隔 L 是可以再增長還是減短 ? (Greedy Method)
  double lastLanding = a[order[0]];   //  第一架飛機的開始安全區間就是他的著陸時間
  for (i = 1; i < n; i++) {       //  依據給定的著陸順序 array order 每一架飛機 , 決定 L 要增加還是減短
    double targetLandingTime = lastLanding + L;                  // targetlandingtime: 預計著陸時間
    if (targetLandingTime <= b[order[i]])                        // 預計著陸時間在安全時間內
      lastLanding = max(a[order[i]], targetLandingTime);   //  計算真正的著陸時間 lastlanding
    else
```

LA 4445 Code (2/3)

```
int main() {
 freopen("1079.in","r",stdin);
 freopen("1079.out","w",stdout);
 while (scanf(“%d”, &n), n) {   //  2 <= n <= 8
   for (i = 0; i < n; i++) {
    scanf("%lf %lf", &a[i], &b[i]);
    a[i] *= 60; b[i] *= 60;          // a[i],b[i]: 輸入的時間單位由分改為秒
    order[i] = i;
   }
```

LA 4445 Code (3/3)

二分法

```
maxL = -1.0;
  do {
    double lo = 0, hi = 86400;
    L = -1;      // 就特定的著陸順序決定時間間隔 L 是要增長還是減短?
    while (fabs(lo - hi) >= 1e-3) {
      L = (lo + hi) / 2.0;                                  Complete
      double retVal = greedyLanding();                      Search
      if (retVal <= 1e-2) lo = L;     // 回傳負值： L 要增長
        else hi = L;                    // 回傳正值： L 要減短
    }    // L: 最小著陸間隔時間
    maxL = max(maxL, L);  // 保留到目前為止最大的最小著陸間隔時間 , 存在 max
  } while (next_permutation(order, order + n));
  maxL = (int)(maxL + 0.5);
  printf("Case %d: %d:%0.2d\n", caseNo++, (int)(maxL/60), (int)maxL%60);
 }          //   輸出由秒單位換成分 : 秒型式
 return 0;
}
```