1

Multi-targets Search

```c
// util.h
typedef struct {
    int n, m;     // 地图的长宽
    int **grid;   // n*m的01数组，1表示墙
    int **food;   // n*m的01数组，1表示这个位置有食物
    int start_x, start_y;
} game_state_t;
```

Greedy

每次寻找距离当前位置最近的target

Enumeration

假设有k个targets，枚举所有k!种寻找顺序

2-opt

1. 构造一组初始解，比如greedy方法产生的解。（用targets的寻找顺序表示一组解
2. 每次随机选择两个不同的targets，反转两个target中间的路线，比如：假设有5个target，初始解是（2，3，5，4，1），此时选择3和1，反转后的解是（2，1，4，5，3）
3. 如果新的解距离总长小于当前解距离总长，则用新解替换当前解。
4. 直到解不能被更新，算法结束。

Simulated annealing

- Let $s = s_0$
- For $k = 0$ through $k_{\max}$ (exclusive):
  - $T \leftarrow$ temperature($k_{\max}/(k+1)$)
  - Pick a random neighbour, $s_{\text{new}} \leftarrow \text{neighbour}(s)$
  - If $P(E(s), E(s_{\text{new}}), T) \geq \text{random}(0, 1)$:
    - $s \leftarrow s_{\text{new}}$
- Output: the final state s

Option: $$P(E(s), E(s_{new}), T) = e^{\frac{(E(s)-E(s_{new}))}{T}}$$

Tests

1. Greedy，2-opt, SA测试数据相同，共11组test cases
2. Enumerate有5组test cases，与上面前5组相同。