(67686) Mathematical Foundations of AI May 28, 2008

Lecture 3

Lecturer: Ariel D. Procaccia *Scribe: Keren Haas, Yair Movshovitz, Dana Attias*

1 Robotic Search

1.1 Offline

Problem definition:

- Input
  - Grid of size $n \times n$. Each cell in the grid is either *free* or *occupied*.
  - Robot starting position and goal position.
- Actions
  - At each step the robot can move to an adjcent (4-neighbours) free cell.
- Output
  - A minimal set of movements between the starting position and goal position that uses only free cells.

Example - In the example shown in figure 1 , $n = 4$ and the quality of the optimal solution is 6 steps. Note that we will use the same legend for all figures in the scribe.

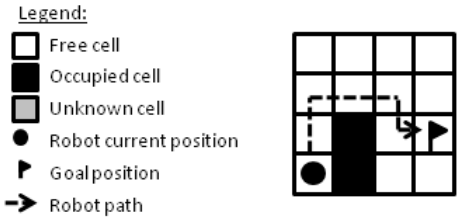

Figure 1: Example of offline robotic search.

Note that it is possible to calculate an optimal path by using BFS or, if the movement between cells has a cost function, by using Dijkstra.

1.2 Online

In this case the problem definition is the same as the offline version with one difference - the status of the cells in the grid is initially unknown. Whenever the robot arrives at a cell it can determine the status of its 4-neighbours.

Example Figure 2 shows two consecutive steps of an online robot search.

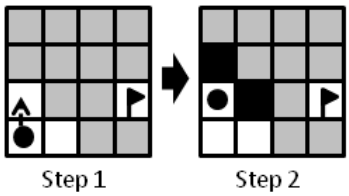

Figure 2: Online robot search example.

Definition 1 (competitive ratio algorithms) *An algorithm* ALG *is said to be* α *- competitive if for every input* I *its performance holds* $\text{ALG(I)} \leq \alpha \cdot \text{OPT(I)}$*, where* OPT *is the optimal solution of the offline problem.*

Theorem 2 *No algorithm has a better (lower) competitive ratio than* $\Omega(n)$*.*

Proof

We define a family of grids in which each even row is blocked from the second cell to the $n-1$ cell, as can be seen in figure 3. This definition creates *corridors* at the odd rows. Each such corridor is blocked at the $n-1$ cell except for one of them. The algorithm needs to try and traverse all the corridors in order to find the one which is not blocked. The number of steps is at least $\frac{n}{2} \cdot (n-2)$, the $\frac{n}{2}$ term is the number of corridors and $(n-2)$ is for the length of each corridor. The optimal solution satisfies $\text{OPT} \leq 2n$ so the ratio is

$$\frac{\frac{n}{2}(n-2)}{2n} = \Omega(n)$$

■

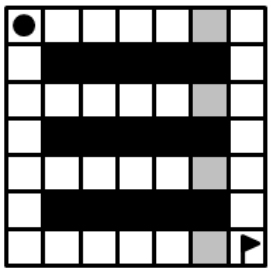

Figure 3: Example of a grid that can be used to prove the lower bound on competitive ratio. Only one of the four unknown cells is actually free. The robot will not know which corridor is free before advancing down that corridor. Hence, in the worst case, the robot must traverse all the corridors in order to find the one that is not blocked.

When using DFS there is no difference between the offline and online problems. Note that DFS has a running time of $O(|V| + |E|)$ steps. In our problem $|E| \leq 4|V|$ (each node has 4 neighbours) , and so we get a running time of $O(|V|) = O(n^2)$.

Algorithm 1 solves the online problem in a simple way using DFS. Figure 4 depicts two examples of local grid areas, similar to the areas that are accessible in each step of the algorithm.

Algorithm 1 Simple algorithm for solving the online problem using DFS

1. Scan all cells within a distance of $n^{\frac{1}{2}}$ from the start position using DFS. If the goal position is found - stop.
2. Same as (1) with a distance of $n^{\frac{3}{4}}$.
3. Scan the entire grid.

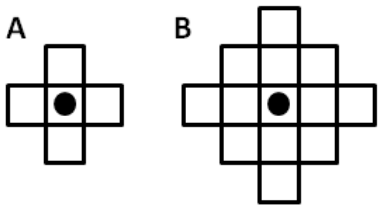

Figure 4: (A) Local grid within a distance of 1 from the start position. (B) Local grid within a distance of 2 from the start position. These cells represent a local area around the start position. Note that some of these cell might be unreachable or occupied.

Claim 3 *Algorithm 1 has a competitive ratio of* $O(n^{1.25})$.

Proof

We examine the following three cases:

1. The goal is reached during step 1 of the algorithm. In this case there are no more than $O((n^{0.5})^2) = O(n)$ cells which the DFS explores, i.e. the algorithm does no more than $O(n)$ steps. The optimal solution takes no less than 1 step, and so the ratio is bounded by $O(n)$.
2. The goal is reached during step 2 of the algorithm. As the goal is found in this step and not in the previous one, the optimal solution is larger than $n^{0.5}$. The DFS takes no more than $O((n^{\frac{3}{4}})^2) = O(n^{1.5})$ steps. When adding the steps taken in the first stage of the algorithm we get $O(n) + O(n^{1.5}) = O(n^{1.5})$. The ratio becomes $\frac{O(n^{1.5})}{n^{0.5}} = O(n)$.
3. The goal is reached during step 3 of the algorithm. As the goal is found in this step and not the previous ones, the optimal solution is larger than $n^{0.75}$. The DFS takes no more than $O(n^2)$ steps. The ratio is $\frac{O(n^2)}{n^{0.75}} = O(n^{1.25})$.

■

Open problem in A.I (challange for an Msc. thesis): Find an upper and lower bound for the competitive ratio of the online robot coverage problem using multiple robots (discussed in the previous lecture).

1.3 D*

This algorithm (shown in algorithm 2) modifies its hypothesis of the shortest path as new information is gathered. An example of several steps in the D* algorithm is shown in figure

5. This algorithm requires a small amount of computing power and memory, and is considered very practical.

Algorithm 2 D*

1. Calculate shortest estimated path using the known data.
2. After each step re-calculate (if needed) the shortest path according to the new data revealed at this step.

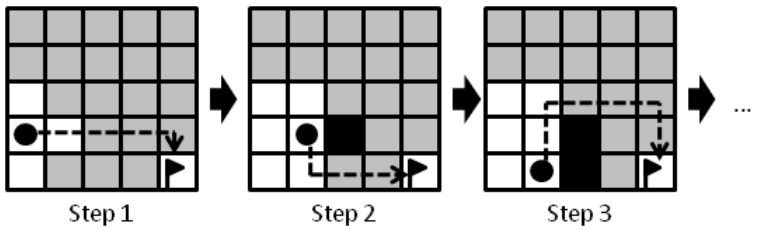

Figure 5: D* example.

Claim 4 *[1] The D* algorithm has a worst-case running time of* $\Omega(\frac{log(n)}{log(log(n))}n^2)$.

We can see that D* in the worst-case is worse than DFS. However, it usually performs better and is much more efficient. We will now further investigate the worst-case behavior of D*.

Theorem 5 *[1] The D* algorithm has a worst-case running time of* $O(n^3)$.

Proof

A *blockage detection (b.d)* occurs when the robot discovers an occupied cell it did not know about in the previous step. Let b be the number of $b.d$ encountered throughout the algorithm. Let c^i be the cell the robot is at, when the i-th $b.d$ is encountered. Let L^i be the number of steps the robot advances between the $(i-1)$-th $b.d$ and the i-th $b.d$.

Once all the blockages are detected, the maximal number of steps to the goal position is, at most, n^2. Hence, the maximal number of steps the robot advances throughout the algorithm is at most

$$\sum_{i=1}^{b} L^i + n^2 \tag{1}$$

Let d_v^i be the estimated distance between cell v and the goal position, after the i-th $b.d$. If the goal is initially unreachable from cell v, we define $d_v^0 = 0$. If after the i-th $b.d$ we discover that the goal is unreachable from cell v, we define $d_v^i = d_v^{i-1}$. For each v, d_v^i is monotonically increasing in i.

We define a set of functions Φ^i

$$\Phi^i = \sum_v d_v^i$$

The Φ^i function is monotonically increasing in i. Moreover, $\Phi^b \leq n^4$ (the sum has n^2 elements, each smaller than n^2).

Let $\Delta^i = d^i_{c^i} - d^{i-1}_{c^i}$ be the difference in estimate caused by the i-th *b.d.* Let D be the current estimated distance between the robot and the goal. The difference in D between the starting position and the b-th *b.d* is

$$\sum_{i=1}^{b} \Delta^i - L^i$$

The difference is also defined as

$$-n^2 \leq d^b_{c^b} - d^0_{c^0} = \sum_{i=1}^{b} \Delta^i - L^i \leq n^2$$

Hence,

$$-n^2 \leq \sum_{i=1}^{b} \Delta^i - L^i$$

$$\sum_{i=1}^{b} L^i \leq \sum_{i=1}^{b} \Delta^i + n^2 \tag{2}$$

Combining our knowledge from equations (1) and (2) results in an upper bound on the number of steps

$$\sum_{i=1}^{b} \Delta^i + 2n^2$$

Lemma 6

$$\sum \Delta^i \leq 2n^3$$

Note that the lemma provides the final step in proving the theorem, since, when using the lemma, the maximal number of steps becomes less than

$$2n^2 + 2n^3 = O(n^3)$$

Proof of Lemma

Let $i \in \{1, \ldots, b\}$. For a given cell c^i, let X_z be a cell in an estimated distance z from c^i after the i-th *b.d.* By the triangle inequality, it holds that

$$d^i_{c^i} \leq d^i_{X_z} + z$$

Therefore,

$$d^i_{X_z} \geq d^i_{c^i} - z \tag{3}$$

At the $(i-1)$-th *b.d* the estimated distance between X_z and c^i is, at most,

$$d^{i-1}_{X_z} \leq z + d^{i-1}_{c^i} \tag{4}$$

From (3) and (4) it holds

$$d^i_{X_z} - d^{i-1}_{X_z} \geq d^i_{c^i} - d^{i-1}_{c^i} + 2z = \Delta^i - 2z$$

and

$$0 \leq d^i_{X_z} - d^{i-1}_{X_z}$$

where the second inequality is derived from the fact that the distances are monotonically increasing with i. Hence,

$$d^i_{X_z} - d^{i-1}_{X_z} \geq \max\{\Delta^i - 2z, 0\} \tag{5}$$

After the i-th $b.d$, for each distance $z = 0, \ldots, d^i_{c^i}$ there is a cell at an estimated distance z from c^i. We denote this cell as X_z. Note that for all $z \neq z'$, $X_z \neq X_{z'}$.

$$\Phi^i - \Phi^{i-1} = \sum_v d^i_v - d^{i-1}_v \geq \sum_{z=0}^{d^i_{c^i}} d^i_{X_z} - d^{i-1}_{X_z} = \Delta^0 + \sum_{z=1}^{d^i_{c^i}} d^i_{X_z} - d^{i-1}_{X_z}$$

From equation (5)

$$\Phi^i - \Phi^{i-1} \geq \Delta^0 + \sum_{z=1}^{d^i_{c^i}} \max\{\Delta^i - 2z, 0\} \geq \Delta^0 + \sum_{z=1}^{\Delta^i} \max\{\Delta^i - 2z, 0\} \geq \frac{(\Delta^i)^2}{4}$$

$$n^4 \geq \Phi^b - \Phi^0 = \sum_{i=1}^{b} \Phi^i - \Phi^{i-1} \geq \sum_{i=1}^{b} \frac{(\Delta^i)^2}{4}$$

We now need to solve:

$$\max \sum \Delta^i$$

$$\text{s.t. } n^4 \geq \sum_{i=1}^{b} \frac{(\Delta^i)^2}{4}$$

$$\forall i, \Delta^i \geq 0$$

Using *Lagrange Multipliers* the maximal sum is given when $\Delta^i = \frac{2n^2}{\sqrt{b}}$ for all i and then

$$\sum \Delta^i = \sum_{i=1}^{b} \frac{2n^2}{\sqrt{b}} = \sqrt{b} 2n^2 \leq 2n^3$$

since b (the number of $b.d$'s) is smaller than n^2 (the number of cells in the grid).

■

■

References

[1] C. Tovey, S. Greenberg, and S. Koenig. Improved analysis of D*. In *Proceedings of ICRA*, pages 3371–3378, 2003.