Task 3. Reinforcement Learning

Garoe Dorta-Perez
CM50245: Computer Animation and Games II

March 18, 2015

1 Introduction

2 A* path finding

The A* algorithm can find paths optimally in a graph based on a heuristic. It works under the assumption that the distance heuristic will never overestimate the cost of the path and the path cost will not decrease as we travel through it. In a maze world, where the movement possibilities are north, south, east and west, with fixed positive costs. Using a Manhattan distance for the actual path cost and a euclidean distance to compute the heuristic will fit the assumptions.

In Algorithm 1 an overview of a general implementation of the A* in pseudocode is given. As stated in the previous paragraph, in our case, graph.cost() and heuristic() returns respectively, the Manhattan and the euclidean distance between two nodes. An example output with a small labyrinth where a path is successfully found is shown in Figure 1.

3 Q learning in noughts and crosses

4 Q learning in BlackJack

5 Results and conclusions

Algorithm 1: A*

Data: *goal* goal position, *start* start position, *graph* graph with the tiles in the map.
Result: *path* path from *start* to *goal*, *cost* total cost of the path.

```
frontier = PriorityQueue();
frontier.put(start, 0);
came_from = {};
cost_so_far = {};
came_from[start] = None;
cost_so_far[start] = 0;
while not frontier.empty() do
    current = frontier.get();
    if current == goal then
        break;
    end
    for next in graph.neighbors(current) do
        new_cost = cost_so_far[current] + graph.cost(current, next);
        if next not in cost_so_far or new_cost <cost_so_far[next] then
            cost_so_far[next] = new_cost;
            priority = new_cost + heuristic(goal, next);
            frontier.put(next, priority);
            came_from[next] = current;
        end
    end
end
path = getPath(came_from);
goal = cost_so_far[current];
```

```
 X 00000000001111111111
   01234567890123456789
Y
00 S*******************
01 .##################*
02 .##..###.#.#.#.###**
03 .##..###.#.#.#.###*.
04 .#....##.#.#****##*.
05 .#..#....#***.#****.
06 .####.....*####.....
07 .########.*.#######.
08 .#........G#........
09 .#.#######..#######.
10 .#....#..#..........
11 .#####.#.#.#####....
12 .#.#.###.#.#.#.###..
13 .#.#.###.#.#.#.###..
14 .#....##.#.#....##..
15 .#..#....#....#.....
16 .####......####.....
17 ..#######...###.####
18 .#.........#........
19 ....................
```

Figure 1: Path finding example, S is the start position, G is the goal position, $*$ are the path points, . are empty spaces and # are walls.

```
---- Policy ----

X = Hit, Blank = Stick

With usable ace     | Player hand value
                    | 21
                    | 20
                    | 19
X   X   X X   X X X | 18
X X X X X X X X X X | 17
X X X X X X X X X X | 16
X X X X X X X X X X | 15
X X X X X X X X X X | 14
X X X X X X X X X X | 13
X X X X X X X X X X | 12
X X X X X X X X X X | 11
1 2 3 4 5 6 7 8 9 10 -> Dealer seen card

Without usable ace  | Player hand value
                    | 21
                    | 20
                    | 19
                    | 18
X               X X | 17
X             X X   | 16
X           X X X X | 15
X           X X X X | 14
X X         X X X X | 13
X X X X   X X X X X | 12
X X X X X X X X X X | 11
1 2 3 4 5 6 7 8 9 10 -> Dealer seen card
```

Figure 2: Blackjack policy example, with 20% exploration rate after 100000 games.

```
---- Policy ----

X = Hit, Blank = Stick

With usable ace     | Player hand value
                    | 21
                    | 20
                    | 19
X   X X X     X X X | 18
X X X X X X X X X X | 17
X X X X X X X X X X | 16
X X X X X X X X X X | 15
X X X X X X X X X X | 14
X X X X X X X X X X | 13
X X X X X X X X X X | 12
X X X X X X X X X X | 11
1 2 3 4 5 6 7 8 9 10 -> Dealer seen card

Without usable ace  | Player hand value
                    | 21
                    | 20
                    | 19
                    | 18
X             X     | 17
X           X X     | 16
X           X X X X | 15
X           X X   X | 14
X           X X X X | 13
X   X       X X X X | 12
X X X X X X X X X X | 11
1 2 3 4 5 6 7 8 9 10 -> Dealer seen card
```

Figure 3: Blackjack policy example, with 40% exploration rate after 100000 games.

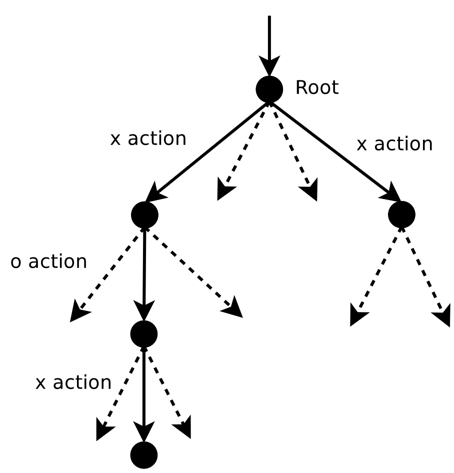

Figure 4: Path finding example, S is the start position, G is the goal position, $*$ are the path points, . are empty spaces and # are walls.