ECE 448: Artificial Intelligence

Assignment 1: Search

Fall 2017

Section: R3 (3 credits)

Jiahe Liu (jliu143)

Haoliang Yue (hyue3)

Ziwei Zhu (zzhu39)

Introduction

In this assignment, we implement a Pacman solver, or maze puzzle solve, using four different general-purpose searching algorithm: Breadth-first Search (BFS), Depth-first Search (DFS), Greedy Best-first Search and A* Search. Tasks contain searching single dot (Part 1.1) and multiple dots (Part 1.2) in provided maze setups. The program is written in Python.

Part 1.1 Single Dot Search

In this first part, the goal is to search for a single dot in a maze puzzle with optimal path possible. All algorithms are applicable to this part.

We have defined a “node” class for digitizing the maze in text file. Each node has class variables: x, y, char, visited, parent, g, h, root, and rank. “x” and “y” are indices for node in a 2D list of digitized maze; “char” stores character of a node; “parent” records the previous node on a path with respect to current node, it is useful when backtracking the solution path; “g” and “h” are path-cost and heuristic of a node; “root” and “rank” are used in Minimum Spanning Tree construction. In our definition, start node is node with character “P”, and goal nodes are node with character “.”.

For each searching algorithm, two parameters are provided: “start” node, and “goal” node, or “goal” list for A* in Part 1.2. Inside algorithms, “front_list” and “visit_list” are initialized to keep tracks of frontiers and visited nodes. “front_list” is constructed as queue in BFS, stack in DFS, and priority queue in A*.

- **Breadth-first Search:**

  I. **State Representation:** State in BFS contains node class variables: x, y, char, parent, and visited. These specs are updated when the node is being explored. Goal count and node expanded are implicitly considered as state representation, but BFS merely uses them.

  II. **Transition Model:** Transitions in BFS are determined by children of a node. At the beginning of an iteration, the program pops out the first node in “front_list", basically a **queue**, and explores it as a node in frontier. All viable children of that node are discovered and pushed into the back of “front_list”, if the child is not a wall and it has not been visited yet. And the program starts a new iteration of transition. During iterations, a variable called “node_exp” keep the record of total node expanded by each search call.

  III. **Goal Test:** Goal Test happens immediately after new frontier been popped. The program checks if frontier is goal node. If true, then it returns the goal node back to its caller to draw solution, and exits the search process.

  IV. **Performance:** BFS guarantees to provide the optimal search result given that step cost is uniform, because BFS is a complete search. On the other side, it requires a relative large amount of space, since it explores the most number of node among all four searching algorithms. Please refer attached screenshots on the last section of the report to see exact results.

- **Depth-first Search:**

  I. **State Representation:** Same as that of BFS.

II. Transition Model: Transitions in DFS are similar to that of BFS. But DFS uses **stack**, instead of queue, to store frontier. During child discovery, DFS pushes viable children up to the front of “front_list”.

III. **Goal Test:** Same as that of BFS.

V. **Performance:** On the opposite of BFS, DFS is neither complete nor optimal. But it requires less space than BFS does, and solves the maze puzzle faster, although with suboptimal solution path. Please refer attached screenshots on the last section of the report to see exact results.

- **Greedy Best-first Search:**

I. **State Representation:** Similar to that of BFS, but Greedy use heuristic as parameter to its evaluation function as well. For this part of the assignment, heuristic is defined as Manhattan distance between two nodes. This heuristic is admissible, because in a 2D square-block maze puzzle, Pacman cannot travel diagonally, so the best possible path between the Pacman and the dot would be their Manhattan distance.

II. **Transition Model:** Greedy search uses same transition model as A* search does, except that Greedy does not require path cost as evaluation parameter. So, the **priority queue** in Greedy Search sort the frontier only based on heuristic.

III. **Goal Test:** Same as that of BFS.

IV. **Performance:** Greedy cannot provide complete and optimal search result. But it is fast and takes less time than A* search. Please refer attached screenshots on the last section of the report to see exact results.

- **A* Search:**

  I. **State Representation:** Similar to that of Greedy, but in A*, both path cost and heuristic are part of state representations. Because A* uses them as its evaluation cost parameters: $f(n) = g(n) + h(n)$.

  II. **Transition Model:** Again, similar to BFS, A* uses **priority queue** to store frontier. At the beginning of each iteration, the program sorts and updates "front_list" to ensure that the node to be popped as least evaluation cost. And then it follows the flow of BFS, until child discovery. At this point, the program not only checks for viable children, but also checks whether current frontier could have lower evaluation cost when approached from a different parent. And it updates node specs, especially **g** and **h** for those who passed the check.

  III. **Goal Test:** Same as that of BFS

  IV. **Performance:** A* is a complete search and it would render optimal result given admissible heuristic. In part 1.1, our Manhattan heuristic is admissible, so A* gives optimal maze puzzle solution path. Also, it uses less space than BFS does. So, in comparison, A* is the most efficient searching algorithm. Please refer attached screenshots on the last section of the report to see exact results.

Part 1.2 Multiple Dots Search

In second part, the goal is to search all dots spreading across a maze puzzle with optimal path possible. Only A* Search is applied to this problem.

Our approach is to modify the heuristic function for A*. Since now Pacman faces multiple dots, it would be unreasonable to choose as single dot as goal node and compute their Manhattan distance. It could be the case that Pacman and the nearest dot are separated by a single wall, but Manhattan heuristic cannot see the wall. So, this heuristic would push Pacman to pursue that dot behind the wall, and it could add a great amount of node to node expanded as well as to solution.

Instead, we have come up with a more globally admissible heuristic. First, we traverse the maze puzzle to find all dot existed, and store them into a goal list. Then we use that list along with start node to build a Minimum Spanning Tree using Union by Rank algorithm. The tree weights in our MST are computed as Manhattan distance. Our heuristic is the sum of all tree weight.

We think this heuristic is almost admissible. As described above, straight forward Manhattan heuristic with respect to a single goal node makes no sense in the case of multiple goal presented. However, a Manhattan distance MST can partially represent the path cost that Pacman is facing. Since the tree reaches all goal with least cost in total, this circumstance would align with the Pacman optimality we are try to achieve. And each time there is a goal node removed from goal list, the MST would be reconstructed and the heuristics would be updated, so we believe this heuristic is almost consistent as well.

In terms of the actual running process, each time A* would reach a single dot using the MST heuristic. Once it returns with reached goal node, the program would compute current path cost based on that goal node, and remove it from goal list. Then A* is invoked again with parameter of last reached goal node and updated goal list, until all goals are reached and goal list is empty. At last the total path cost and total node expanded would be presented along with the goal node labeled with reached sequence. The search outcomes are attached on the last section of the report.

Individual Contribution

Jiahe Liu	A* Search in 1.1, heuristic in 1.2
Haoliang Yue	Greedy Search, A* Search in 1.2
Ziwei Zhu	BFS, DFS, Assignment Report

Note

Please enter file index when run the program, index definition can be function in #MainFunction# and change the #SearchAlgorithm# part to run different algorithm.

Thanks for your understanding.

Solution Screenshots

Figure 1: mediumMaze using BFS

Figure 2 bigMaze using BFS

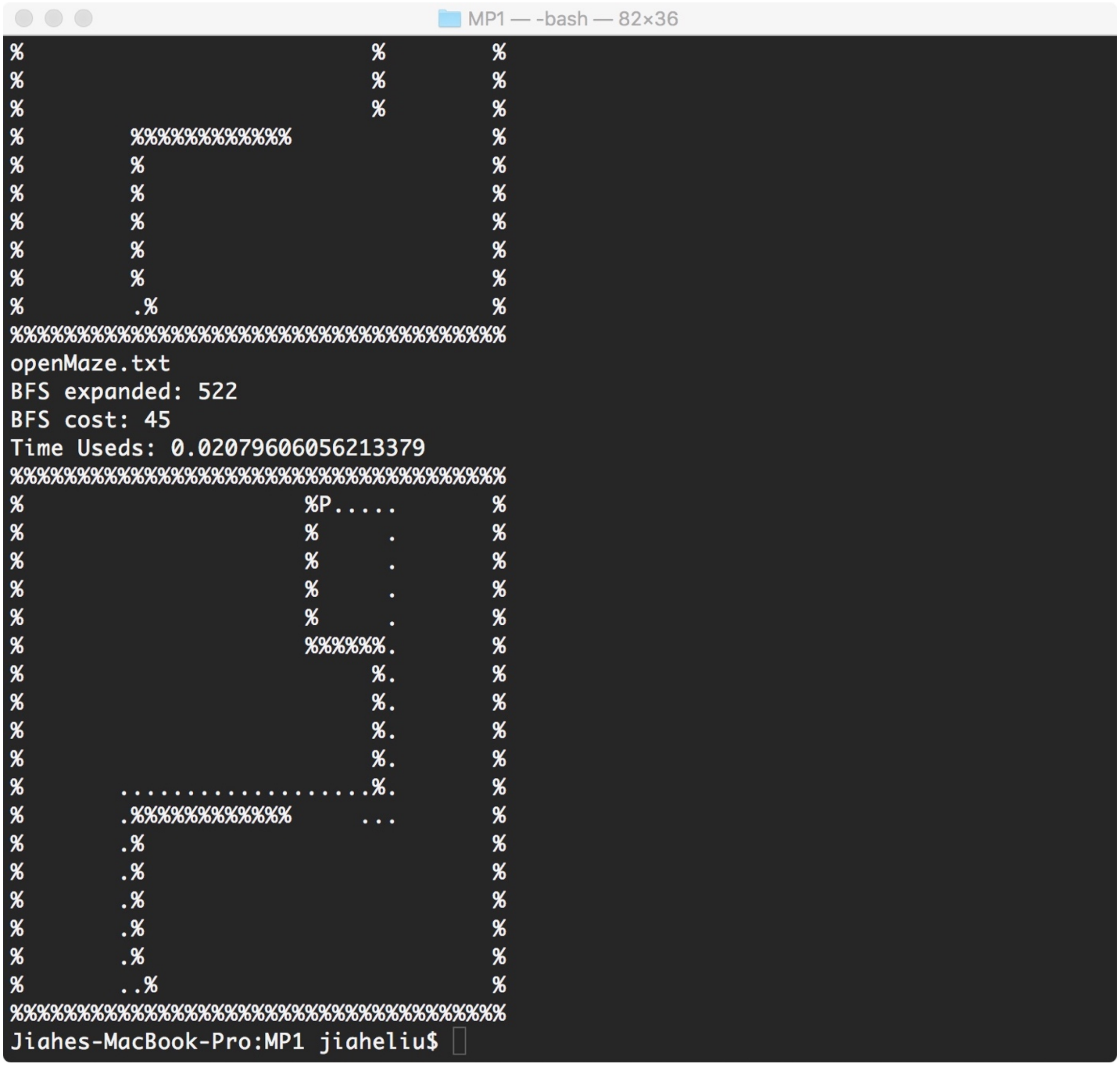

Figure 3 openMaze using BFS

```
% %   %   % %   %   % %   % %     %   %   %     % %   %   % %
%   %%% % % % % % % % % %%% % %%%%%%% %%% % %%% % % %%%%%%% %
% %   % %   % %   %   % %                 % %   % % %       %
% %%% %%% %%% %%% % %%% % %%% % %%%%%% %%%% % %%%%% % % %%% %
%   %         %   % %   % %   %         %   % %     % % %   %
%%% %%% % %%%%% %%%%% %%%%% %%% %%%%%%% % %%% % %%%%%%% % %%%
%P    %       %             %         %     %           %   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mediumMaze.txt
DFS expanded: 281
DFS cost: 204
Time Useds: 0.011400938034057617
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%         % %           %     %           %           %    .%
% %%%% %% % % %%%%% %%%%% %%% % %%%% %% %%% %% %% %%% % % %.%
% %     %   %     %       %   %       %     %   % % %   % %.%
% % %%% % %%%% %% %%%%%%%%% %%%%% %%% % %%% % % % % %%%%% %.%
% %   %   %       %       %       %   %...% % % %     %   %.%
% % %%% %%% %%% %%% % %%% %%%% %% % %%%.%.%%% % %%% %%% %%%.%
%   %     %     %   % %         % % %...%.... %   % %   % %.%
% %%% %%% %%%%% %%%%% % %%% %%% %%%%%.%%%%%%.%%%% %%% %%% %.%
%     %       %         % % %.........%...% ....%     %   %.%
% %%%%% %%%%% %%%%% %%% % % %.%%%%%%%%%.%.% %%%.%%% %%% %%%.%
% %   % %   %         % %   %.%     %...%.% %...        %...%
% % %%% % %%% %%%%% % % %%%%%.% % %%%.% %.%%%.%%%%% %%%%%.%%%
%   %   % %       % % %   %...% %   %.% %.....%     %...%.% %
% %%% %%% % % %%% %%% %%% %.%%% %%% %.% %%%%%%%%% %%%.%.%.% %
% %   %   % %...%   % %   %.%     %  .%   %.....% %...%...% %
%...%%% % % %.%.% % % % %%%.% %%%%%%%.%%% %.%%%.% %.%%%%%%% %
%.%...% %   %.%...%   % %  ... .......    %.%...% %.%       %
%.%%%.%%% %%%.%%%.% %%% % %%%.%.%%%%%% %%%%.%.%%%%%.% % %%% %
%...%... .....%...% %   % %...%.........%...%.%.....% % %   %
%%%.%%%.%.%%%%%.%%%%% %%%%%.%%% %%%%%%%.%.%%%.%.%%%%%%% % %%%
%P..  %...    %.............%         %...  %...        %   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

Figure 4 mediumMaze using DFS

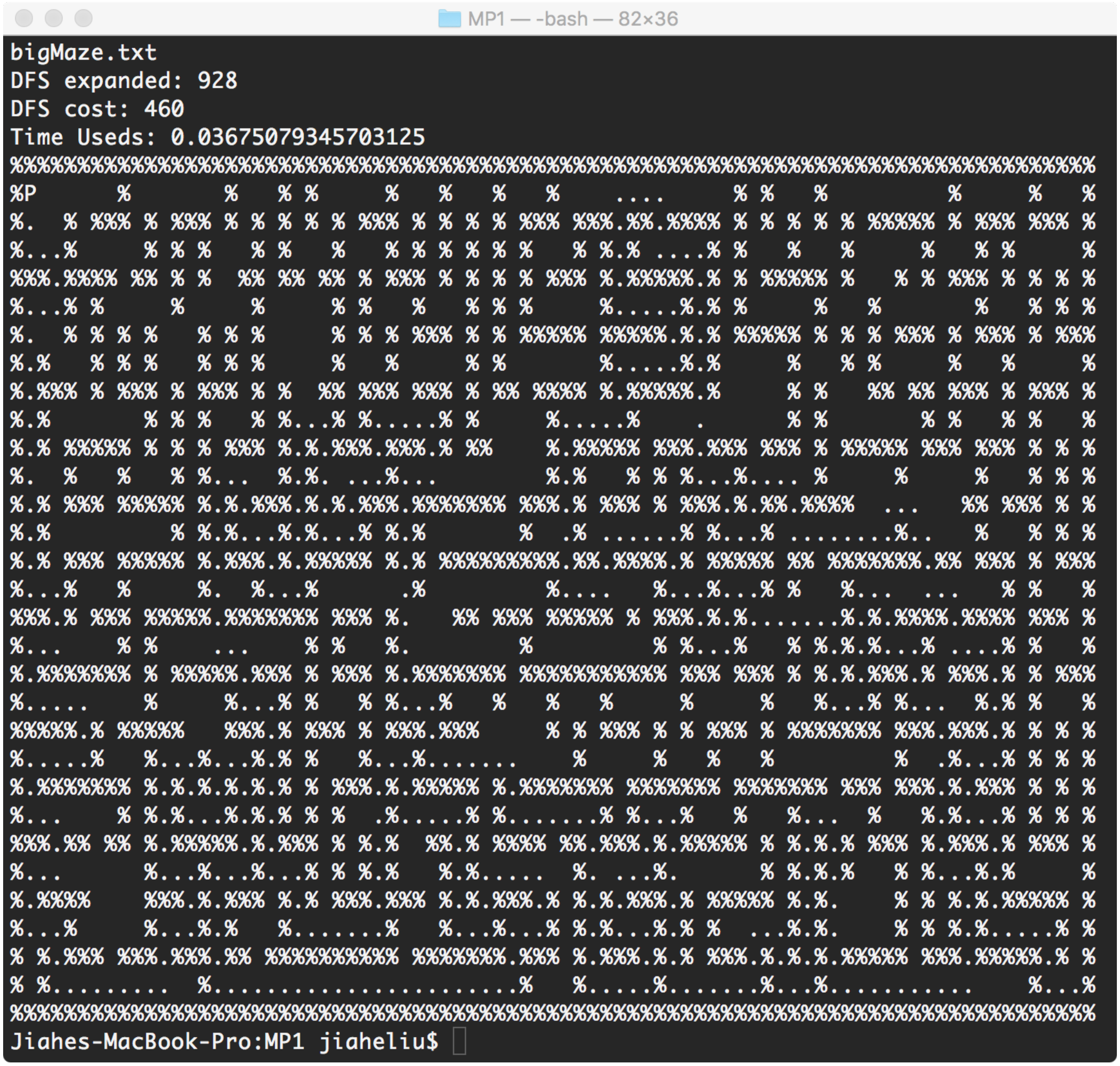

Figure 5 bigMaze using DFS

MP1 — -bash — 82×36

```
%                          %        %
%                          %        %
%                          %        %
%        %%%%%%%%%%%%               %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%        .%                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
openMaze.txt
DFS expanded: 295
DFS cost: 59
Time Useds: 0.012656927108764648
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                     %P            %
%                     %.            %
%                     %.            %
%                     %.            %
%                     %......       %
%                     %%%%%%.       %
%                          %.       %
%                          %.       %
%                          %.       %
%                          %.       %
%.....................     %.       %
%.       %%%%%%%%%%%%........       %
%.       %                          %
%.       %                          %
%.       %                          %
%.       %                          %
%.       %                          %
%.........%                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

Figure 6 openMaze using DFS

MP1 — -bash — 82×36

```
% %   %   % %   %   % %   % %     %   %   %     % %   %   % %
%   %%% % % % % % % % % %%% % %%%%%%% %%% % %%% % % %%%%%%% %
% %   % %   % %   %   % %                 % %   % % %       %
% %%% %%% %%% %%% % %%% % %%% % %%%%%% %%%% % %%%%% % % %%% %
%   %         %   % %   % %   %         %   % %     % % %   %
%%% %%% % %%%%% %%%%% %%%%% %%% %%%%%%% % %%% % %%%%%%% % %%%
%P    %       %             %         %     %           %   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mediumMaze.txt
Greedy expanded: 133
Greedy cost: 118
Time Useds: 0.009923696517944336
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%.........% %           %     %     ....  %...........%.....%
%.%%%% %%.% % %%%%% %%%%% %%% % %%%%.%%.%%%.%% %% %%%.%.% % %
%.%     %.  %     %       %   %  .... %.....%   % % %...% % %
%.% %%% %.%%%% %% %%%%%%%%% %%%%%.%%% % %%% % % % % %%%%% % %
%.%   %...%       %  .....%   ....%   %   % % % %     %   % %
%.% %%%.%%% %%% %%% %.%%%.%%%%.%% % %%% % %%% % %%% %%% %%% %
%.  %  ...%     %   %.%  ...... % % %   %     %   % %   % % %
%.%%% %%%.%%%%% %%%%%.% %%% %%% %%%%% %%%%%% %%%% %%% %%% % %
%.    %  .....%    ...  % % %         %   %     %     %   % %
%.%%%%% %%%%%.%%%%%.%%% % % % %%%%%%%%% % % %%% %%% %%% %%% %
%.%   % %   %.......  % %   % %     %   % % %           %   %
%.% %%% % %%% %%%%% % % %%%%% % % %%% % % %%% %%%%% %%%%% %%%
%.  %   % %       % % %   %   % %   % % %     %     %   % % %
%.%%% %%% % % %%% %%% %%% % %%% %%% % % %%%%%%%%% %%% % % % %
%.%   %   % %   %   % %   % %     %   %   %     % %   %   % %
%.  %%% % % % % % % % % %%% % %%%%%%% %%% % %%% % % %%%%%%% %
%.%   % %   % %   %   % %                 % %   % % %       %
%.%%% %%% %%% %%% % %%% % %%% % %%%%%% %%%% % %%%%% % % %%% %
%...%         %   % %   % %   %         %   % %     % % %   %
%%%.%%% % %%%%% %%%%% %%%%% %%% %%%%%%% % %%% % %%%%%%% % %%%
%P..  %       %             %         %     %           %   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

Figure 7 mediumMaze using Greedy

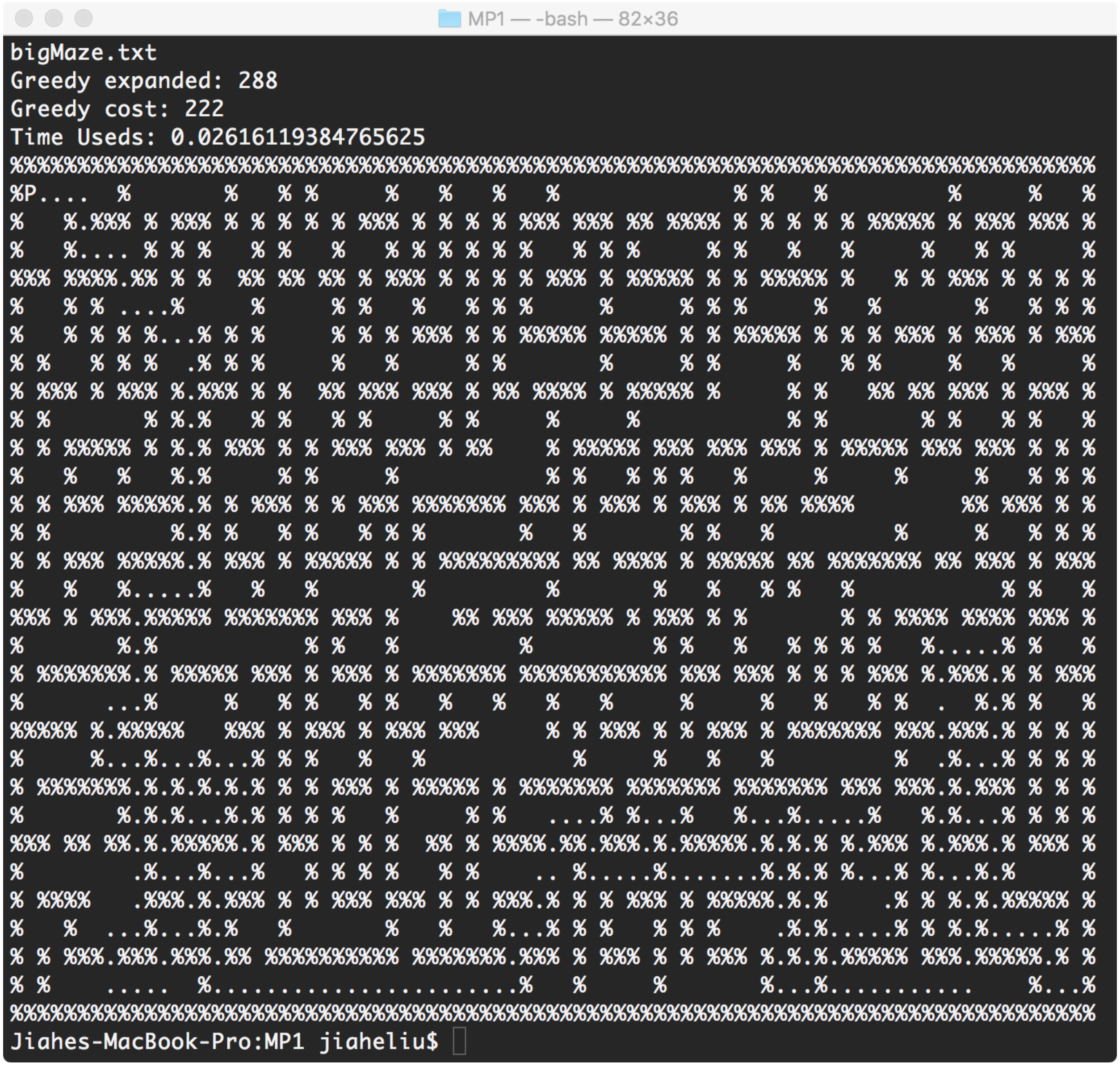

Figure 8 bigMaze using Greedy

```
MP1 — -bash — 82×36
%                          %        %
%                          %        %
%                          %        %
%        %%%%%%%%%%%%               %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%        .%                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
openMaze.txt
Greedy expanded: 154
Greedy cost: 57
Time Useds: 0.009949922561645508
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                     %P            %
%                     %.            %
%                     %.            %
%                     %.            %
%                     %......       %
%                     %%%%%%.       %
%                          %.       %
%                          %.       %
%                          %.       %
%                          %.       %
%       ..............     %.       %
%       .%%%%%%%%%%%%.      .       %
%       .%           .      .       %
%       .%           .      .       %
%       .%           .      .       %
%       .%           .      .       %
%       .%           .      .       %
%       ..%          ........       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

Figure 9 openMaze using Greedy

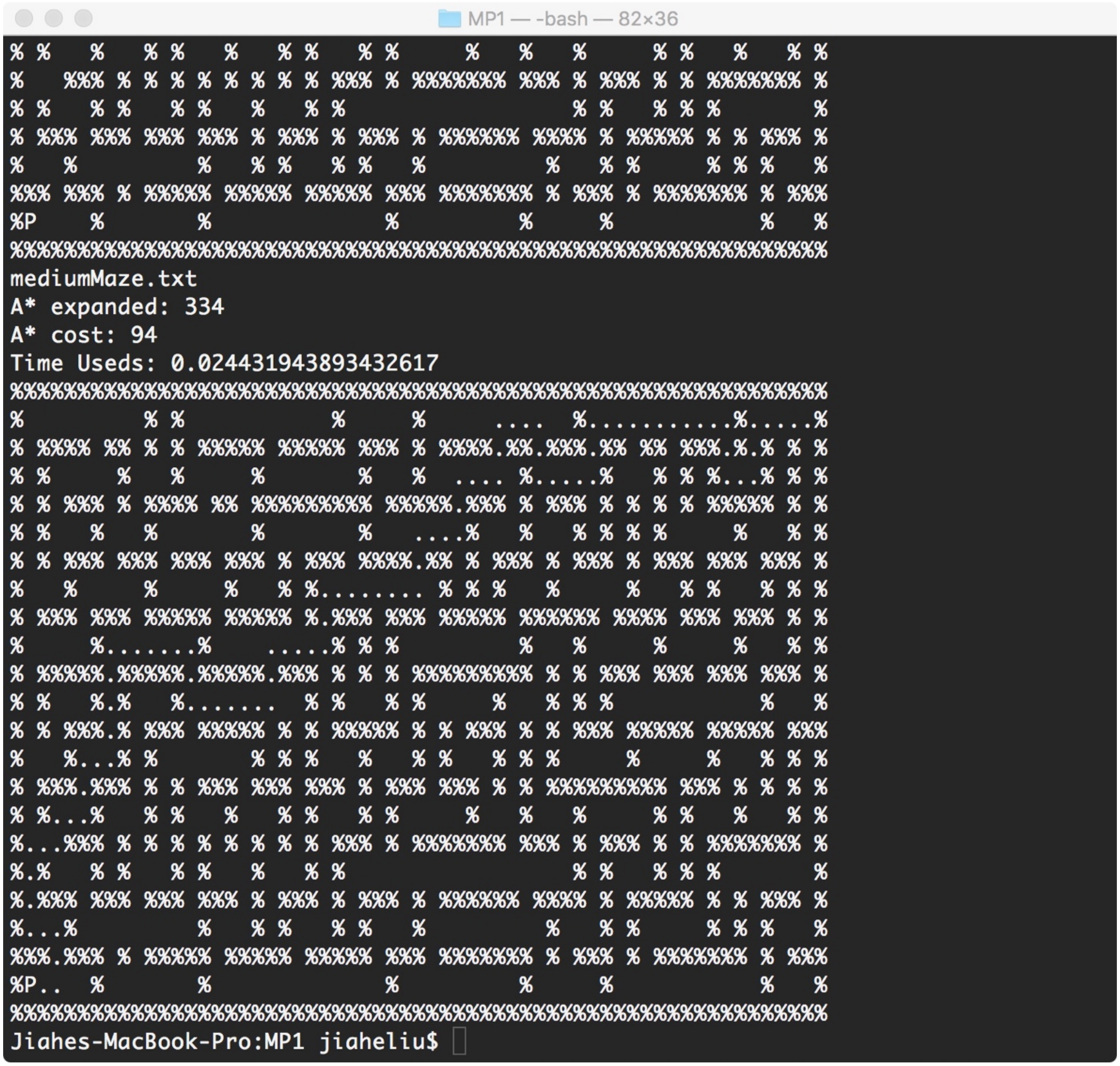

*Figure 10 mediumMaze using A**

*Figure 11 bigMaze using A**

```
%                          %        %
%                          %        %
%                          %        %
%        %%%%%%%%%%%%               %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%        %                          %
%         .%                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
openMaze.txt
A* expanded: 237
A* cost: 45
Time Useds: 0.015826940536499023
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                     %P            %
%                     %.            %
%                     %.            %
%                     %.            %
%                     %......       %
%                     %%%%%%.       %
%                          %.       %
%                          %.       %
%                          %.       %
%                          %.       %
%       ..............     %.       %
%       .%%%%%%%%%%%%........       %
%       .%                          %
%       .%                          %
%       .%                          %
%       .%                          %
%       .%                          %
%       ..%                         %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

*Figure 12 openMaze using A**

```
MP1 — -bash — 82×36
Jiahes-MacBook-Pro:MP1 jiaheliu$ python MP1.py
Modules Imported!
Enter a file index:
3

Original:
%%%%%%%%%%
%.  % .  %
% %.% %% %
% %   .%.%
% .%P%   %
%.  .  . %
% %%%% % %
%. .   %.%
%%%%%%%%%%
A* cost: 40
A* node expanded: 66
Time Useds: 0.0619199275970459
%%%%%%%%%%
%8..%.6  %
%.%7%.%% %
%.%...5%3%
%.9%P%...%
%a. 1..2.%
%.%%%% %.%
%b.c   %4%
%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

*Figure 13 tinySearch using A**

MP1 — -bash — 82×36

```
Jiahes-MacBook-Pro:MP1 jiaheliu$ python MP1.py
Modules Imported!
Enter a file index:
4

Original:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      P      .%      .  .   %
%   %%%%% %%%%%% %  % % %%   %
%   %. %     %   %  % %  %  .%
%.     %     .   %.   %  %%%%%
%%%%% %%%% %%%  %%%%%%%     .%
%.               .    % %%%  %
%% %%%%%%%% %%%%%%%%%%% %    %
%       %               % %%%%
%       %%%%%     %.         %
%.% %%%    %  %   %% %% %%%%%%
%          % .%             .%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A* cost: 165
A* node expanded: 459
Time Useds: 0.33054590225219727
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      P......1%......5..6...%
%   %%%%%.%%%%%%.%..% %.%%  .%
%   %b % ....% ..%..% %. %  7%
%c.... %    .2.. %4.  %. %%%%%
%%%%%.%%%% %%%..%%%%%%%.....8%
%d..........  ...3    % %%% .%
%%.%%%%%%%%.%%%%%%%%%%% %....%
% .     %  .............%.%%%%
%..     %%%%%.....%9......   %
%e% %%%    % .%   %% %%.%%%%%%
%          % a%        .....f%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

*Figure 14 smallSearch using A**

MP1 — -bash — 82×36

```
Jiahes-MacBook-Pro:MP1 jiaheliu$ python MP1.py
Modules Imported!
Enter a file index:
5

Original:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% .  %        .%             %  .%     % %   %. %
%    %%%% %%%%%% %  % % %%         %%%%  %  %%% %
%    %.%    % .% %  % %  %   %   %          %   %
%   .            %.     %%%% %   %% .%%% %      %
% %%%%% %%%%%%  %%%% %           .%    %. %%%%%%%
%.      %   .%       .% %%%  %%%%%%  % %  %   % %
%%% %%  % %%%%%%% %%%%% %    %  . %% % %    %.% %
%    %  %      %  %      P% %%       %   %%%% % %
%.      %   %  %  %     %          %       .%   %
% % %%%    % %    %% %% %%% %% %  % %%%%%%%%%%% %
%   %.   %    %              %    %        .    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A* cost: 240
A* node expanded: 702
Time Useds: 1.378525972366333
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% f  %   .....k%             %  3%     % %   %9.%
% .  %%%%.%%%%%% %  % % %%      ...%%%%  %  %%%.%
% .  %d%..  %.c% %  % %  %   %  .%..........%...%
%...e..........  %a.... %%%% %  .%%.2%%%.%......%
%.%%%%%.%%%%%%. %%%%.%...........4% .  %5.%%%%%%%
%g..   .%...j%.......b% %%%  %%%%%% .% %..%...% %
%%%.%% .%.%%%%%%% %%%%% %    %..1.%%.% %....%7% %
%  . % .%.     %  %      P% %%.  ....%  .%%%%.% %
%h......%.  %  %  %     %......    %    ...6%...%
% % %%%... % %    %% %% %%% %% %  % %%%%%%%%%%%.%
%   %i.. %    %              %    %        8....%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Jiahes-MacBook-Pro:MP1 jiaheliu$
```

*Figure 15 mediumSearch using A**