

# Artificial Intelligence, Spring 2017

## Homework 2 – Search

Xinglu Wang 3140102282 ISEE 1403, ZJU

### 1 Problem 1

Formulate the problem as a search problem,  $M = \{0, 1, 2, 3\}$  denotes the number of missionaries on the left side of river,  $C = \{0, 1, 2, 3\}$  denotes cannibals on the left side,  $B = \{L, R\}$  denotes which side boat parks.

- Start state:  $(3, 3, L)$
- Goal Test: Is state  $== (0, 0, R)$
- State Space: All possible tuple  $(M, C, B)$ , where  $(M = C) \wedge (M = 0) \wedge (M = 3)$
- Successive function:
  - action: from  $(M, C, B)$  to  $(M', C', B')$ , satisfying

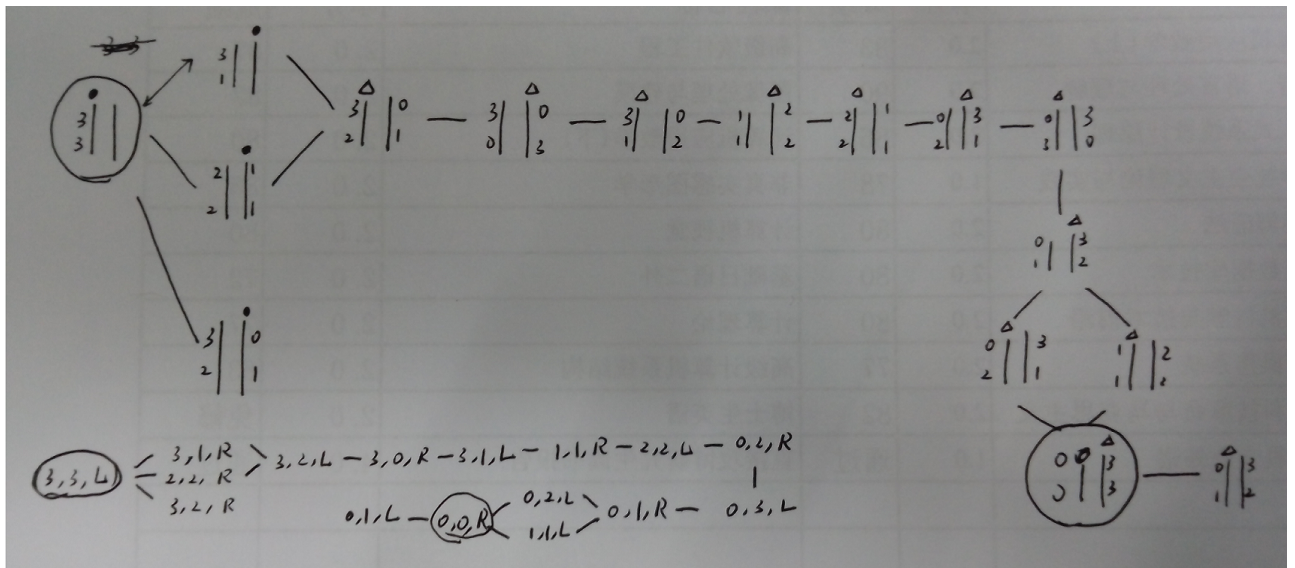
$$M' = M - 1 \vee M' = M - 2 \vee C' = C - 1 \vee C' = C - 2 \vee \begin{cases} C' = C - 1 \\ M' = M - 1 \end{cases}$$

$$B' = \begin{cases} L, & B = R \\ R, & B = L \end{cases}$$

$$(M' = C') \wedge (M' = 0) \wedge (M' = 3)$$

- costs: always 1.

The complete state space:



### 2 Problem 2

- a). If all costs is 1, the cheapest solution is exactly the shallowest solution, i.e.  $g(n) = \text{depth}(n)$ . So uniform-cost search will become breadth-first search.

- b). For best-first search, if  $f(n) = -\text{depth}(n)$ , then it will choose the deepest from frontier first and become depth-first search
- c). If the heuristic function of  $A^*$  is  $h(n) = 0$ , then it will become uniform-cost search.

### 3 Problem 3

- a). branching factor  $b = 4$ .
- b). Since a step means  $|x|$  or  $|y|$  increase by 1, the final state satisfies  $|x| + |y| = k \wedge |x| \leq k \wedge |y| \leq k$ , which has  $4k$  distinct states.
- c). Notation: frontier = set of nodes in frontier;  $|\text{frontier}|$  = number of nodes in frontier; Acc.  $|\text{frontier}|$  = number of nodes expanded. The  $-1$  in tableau shows that I think the destination node will be 'unfortunately' evaluated at the last among the nodes with the same depth, do not need to be expand (just evaluate/test).

depth	frontier	$ \text{frontier} $	Acc. $ \text{frontier} $
$d = 0$	$\{(0, 0)\}$	1	$1 - 1$
$d = 1$	$\{(0, 1), (1, 0), (-1, 0), (0, -1)\}$	4	$5 - 1$
$d = 2$	$\{(-1, 1), (1, 1), (0, 0), (0, 2), \dots, \dots, \dots\}$	16	$21 - 1$
...	...	...	
$d = k$	...	$4^k$	$(\sum_{n=0}^k 4^n) - 1$

Therefore, Acc.  $|\text{frontier}| = -1 + \sum_{n=0}^k 4^n = (4^{k+1} - 4)/3 = (4^{x+y+1} - 4)/3$

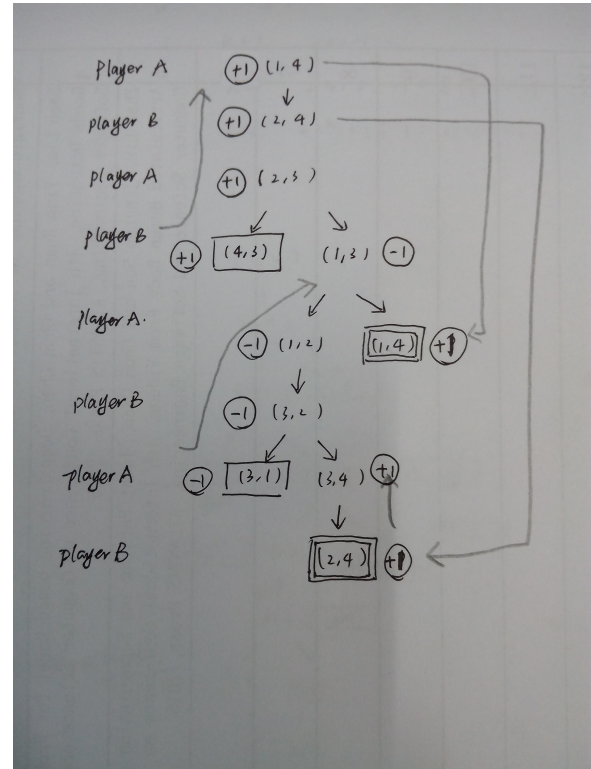
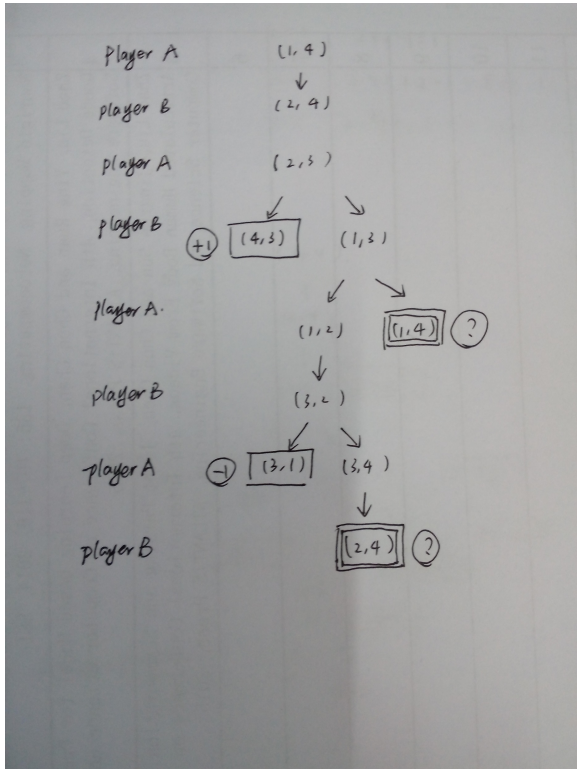
- d). Similarly,

depth	frontier	$ \text{frontier} $	Acc. $ \text{frontier} $
$d = 0$	$\{(0, 0)\}$	1	$1 - 1$
$d = 1$	$\{(0, 1), (1, 0), (-1, 0), (0, -1)\}$	4	$5 - 1$
$d = 2$	$\{(x, y) \in \mathbb{R}^2;  x  +  y  = 2 \wedge  x  \leq 2 \wedge  y  \leq 2\}$	8	$13 - 1$
...	...	...	
$d = k$	$\{(x, y) \in \mathbb{R}^2;  x  +  y  = k \wedge  x  \leq k \wedge  y  \leq k\}$	$4k$	$-1 + \sum_{n=0}^k 4n$

Therefore, Acc.  $|\text{frontier}| = -1 + \sum_{n=0}^k 4n = 2k(k+1) = 2(x+y)(x+y+1)$

- e). Yes,  $h(n) = h^*(n)$  in this case and is admissible.
- f).  $A^*$  will search the nodes 'towards' destination, i.e.  $\{(a, b) \in \mathbb{R}^2; 0 \leq a \leq x \wedge 0 \leq b \leq y\}$ .  
Totally,  $(x+1)(y+1) - 1$
- g). Yes,  $h(n) \leq h^*(n)$ , since some links are removed.
- h). No,  $\exists n, h(n) > h^*(n)$ , if current node  $n$  is one of the ends of the added link.

## 4 Problem 4



**Left:** figure a for problem a. **Right:** figure b for problem b with backed-up values.

- b. Ref. to fig. b. Use back-up, we can get optimal value of all the parent nodes. As for "?" values, we copy the value from visited node. It is right because for player A,  $\max(+1, ?) = +1$ ; for player B  $\min(-1, ?) = -1$  and thus we can prune the infinite branch.
- c. Minimax algorithm fails because it use deep-first tree search and can go into infinite loop. The modified algorithm in b. can not always give optimal decision for all games with loop. For example, if this 4-square game becomes a  $2 \times 2$  matrix game, initialled as
- |   |   |
|---|---|
| A |   |
|   | B |
- , and we define

A reaching	getting value of
(0,0)	1
(0,1)	2
(1,0)	3
(1,1)	4

and B of the opposite value -1,-2,-3,-4.

Then we cannot decide the value of  $\max(+1, ?)$ ,  $\max(+2, ?)$ ,  $\min(-1, ?)$  and so on and especially  $\min(?, ?)$  and  $\max(?, ?)$ . Therefore, just to copy from appeared value will fail.

- d. First consider n is odd.

**Proposition 4.1.** *If n is odd, then there exists a optimal strategy for B to win.*

*Proof.* 1). For  $n = 3$ , B will definitely win.

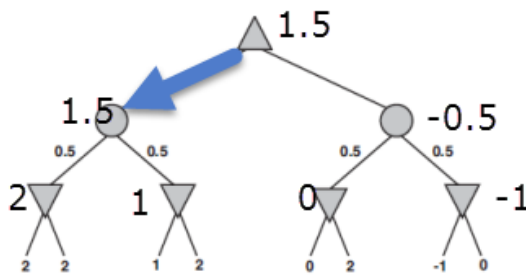
2). Assume  $n-2$  is odd, and there exists a optimal strategy for B to win.

3). For  $n > 3$ , after first round,  $(S_A, S_B) = (2, n-1)$ , the size of square is  $n-2$ . According to 2), there exists a strategy, s.t.  $S_B = 2 \wedge S_A \leq n-2$  after the round of B. Next round of A,  $S_A \leq n-1$  then next round of B, B can choose move left (s.t.  $S_B = 1$ ) as its strategy and win. By induction, B will win when  $n$  is odd.  $\square$

Similarly, if  $n$  is even, then there exists a optimal strategy for A to win.

## 5 Problem 5

a. Up Triangle means max player, and Down Triangle means min player. Note that circle means *stochastic* strategy set. Use back-up, we get



b. Given leaves 1-6, we need to evaluate  $x_7$  and  $x_8$ . If  $\min(x_7, x_8) > 3$  then the best move for max player won't be left. Given leaves 1-7, we do not need to evaluate  $x_8$ .  $0.5 \min(-1, x_8) + 0.5 * 0 \leq -0.5$ , the optimal move for max player keeps left.

c. The value range for left-hand chance node:  $[0, 2]$

d. After evaluating  $x_5$  the value range for right-hand chance node is  $[-2, 0]$ , definitely smaller than 1.5. Thus  $x_6, x_7, x_8$  can be pruned.

