# CS 461 – ARTIFICIAL INTELLIGENCE

## HOMEWORK # 1

**Group Name :** Puzzle_Busters

**Group Members :**

Muhammad Ramish Saeed

Mohamed Aymen Abidi

Kamel Ben Kmala

Yassine Gazzah

Dorra Elmekki

**a. Formulate this puzzle as a state space search. (This part -- that is, part a -- is for you to ponder about. Your answers should appear as comments in the beginning of your code.)**

Firstly we will represent a state as a tuple **(x,y)** where **x** represents the amount of water in the **10-liter** jug and **y** represents the amount of water in the **6-liter** jug.

Also, we will use the following notations for the jugs :

- 'A' for the 10-liter jug
- 'B' for the 6-liter jug

## - What are the states ?

There are **(7*11)** possible states but only **16** are allowed in our case. In fact, some states are not authorized due to the constraints when using the operators (fill A, dump B ...)
The **16** states are :

(0,0) (10,0) (0,6) (6,0) (6,6) (10,2) (0,2) (2,0)
(2,6) (8,0) (8,6) (10,4) (0,4) (4,0) (4,6) (10,6)

## - What are the initial and goal states ?

- Initial state : (0,0)
- Goal state : (8,y) where **y** in {0, 2, 4, 6}

## - What are the operators ?

- **Fill A** : (x,y) <-- (10,y), x < 10
- **Fill B** : (x,y) <-- (x,6), y < 10
- **Dump A** : (x,y) <-- (0,y), x > 0
- **Dump A** : (x,y) <-- (x,0), y > 0
- **Pour A to B** : **if** ( ( x > 0 ) and ( x <= ( 6 - y ) ) ) **then** ( x = 0) **and** ( y = y + x )
  **if** ( ( x > 0 ) and ( x > ( 6 - y ) ) ) **then** ( x = x - ( 6 – y ) ) **and** ( y = 6 )
  ➡ **if** ( x > 0 ) **then** x = max (0, x - ( 6 – y ) ) **and** y = min (6, y + x )

- **Pour B to A** : **if** ( ( y > 0 ) and ( y <= ( 10 – x ) ) ) **then** ( x = x + y ) **and** ( y = 0 )
  **if** ( ( y > 0 ) and ( y > ( 10 – x ) ) ) **then** ( x = 10 ) **and** ( y = y - ( 10 – x ) )
  ➡ **if** ( y > 0 ) **then** x = min (10, x + y ) **and** y = max (0, y - ( 10 – x ) )

## - What is the branching factor ?

The branching factor is the number of the children at each node. If the value is not uniform an average branching factor can be calculated.

➡ The average branching factor is : **ABF = 58/16 = 3.625**

**b. Run your search program 5 times and let it list a path from the initial state to the goal state for each run.**

Test N°1

```
Enter the initial value of a (the big jug) : 0
Enter the initial value of b (the small jug) : 0
Starting the Non-deterministic search...
start     : (0, 0)
Fill A    : (10, 0)
Pour A->B : (4, 6)
Fill B    : (4, 6)
Pour A->B : (0, 6)
Pour B->A : (6, 0)
Fill B    : (6, 6)
Pour B->A : (10, 2)
Dump A    : (0, 2)
Pour B->A : (2, 0)
Fill B    : (2, 6)
Pour B->A : (8, 0)
```

Test N°2

```
Enter the initial value of a (the big jug) : 0
Enter the initial value of b (the small jug) : 0
Starting the Non-deterministic search...
start     : (0, 0)
Fill A    : (10, 0)
Pour A->B : (4, 6)
Fill B    : (4, 6)
Dump B    : (4, 0)
Pour A->B : (0, 4)
Fill A    : (10, 4)
Pour A->B : (8, 6)
```

Test N°3

```
Enter the initial value of a (the big jug) : 0
Enter the initial value of b (the small jug) : 0
Starting the Non-deterministic search...
start     : (0, 0)
Fill B    : (0, 6)
Pour B->A : (6, 0)
Fill B    : (6, 6)
Pour B->A : (10, 2)
Dump A    : (0, 2)
Pour B->A : (2, 0)
Fill B    : (2, 6)
Pour B->A : (8, 0)
```

## Test N°4

```
Enter the initial value of a (the big jug) : 0
Enter the initial value of b (the small jug) : 0
Starting the Non-deterministic search...
start     : (0, 0)
Fill B    : (0, 6)
Fill A    : (10, 6)
Dump B    : (10, 0)
Pour A->B : (4, 6)
Fill B    : (4, 6)
Dump B    : (4, 0)
Pour A->B : (0, 4)
Fill A    : (10, 4)
Pour A->B : (8, 6)
```

## Test N°5

```
Enter the initial value of a (the big jug) : 0
Enter the initial value of b (the small jug) : 0
Starting the Non-deterministic search...
start     : (0, 0)
Fill A    : (10, 0)
Fill B    : (10, 6)
Dump A    : (0, 6)
Pour B->A : (6, 0)
Fill B    : (6, 6)
Pour B->A : (10, 2)
Dump A    : (0, 2)
Pour B->A : (2, 0)
Fill B    : (2, 6)
Pour B->A : (8, 0)
```