ITMO UNIVERSITY

How to Win Coding Competitions: Secrets of Champions

Week 5: Algorithms on Graphs 1
Lecture 6: Introduction to dynamic programming

Maxim Buzdalov
Saint Petersburg 2016

Fibonacci numbers: Definition

$$
\begin{aligned}
F_0 &= 1 \\
F_1 &= 1 \\
F_k &= F_{k-1} + F_{k-2}
\end{aligned}
$$

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

- Assume all $\Theta(1) = 1$

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

- Assume all $\Theta(1) = 1$
- $T_k + 1 = (T_{k-1} + 1) + (T_{k-2} + 1)$

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

- Assume all $\Theta(1) = 1$
- $T_k + 1 = (T_{k-1} + 1) + (T_{k-2} + 1)$
- $T_k = 2 \cdot F_k - 1$

Fibonacci numbers: Definition

$$F_0 = 1$$
$$F_1 = 1$$
$$F_k = F_{k-1} + F_{k-2}$$

How fast is it?

$$T_0 = \Theta(1)$$
$$T_1 = \Theta(1)$$
$$T_k = \Theta(1) + T_{k-1} + T_{k-2}$$

How to compute?

```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

- Assume all $\Theta(1) = 1$
- $T_k + 1 = (T_{k-1} + 1) + (T_{k-2} + 1)$
- $T_k = 2 \cdot F_k - 1 = \left[\frac{1.618\ldots^k}{\sqrt{5}}\right] \cdot 2 - 1$

Fibonacci numbers: Definition

$$
\begin{aligned}
F_0 &= 1\\
F_1 &= 1\\
F_k &= F_{k-1} + F_{k-2}
\end{aligned}
$$

How fast is it?

$$
\begin{aligned}
T_0 &= \Theta(1)\\
T_1 &= \Theta(1)\\
T_k &= \Theta(1) + T_{k-1} + T_{k-2}
\end{aligned}
$$

How to compute?

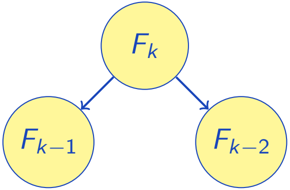
```
int fib(int k) {
    return k <= 1
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

How large is T_k?

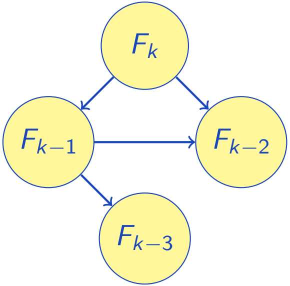
- Assume all $\Theta(1) = 1$
- $T_k + 1 = (T_{k-1} + 1) + (T_{k-2} + 1)$
- $T_k = 2 \cdot F_k - 1 = \left[\frac{1.618\ldots^k}{\sqrt{5}}\right] \cdot 2 - 1$
- That is, exponentially slow in k

Computation of Fibonacci numbers as a graph

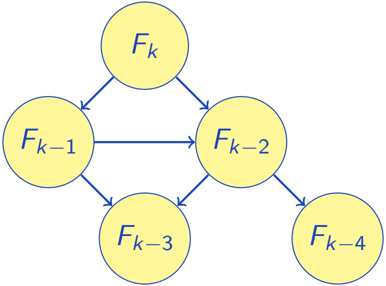
Computation of Fibonacci numbers as a graph



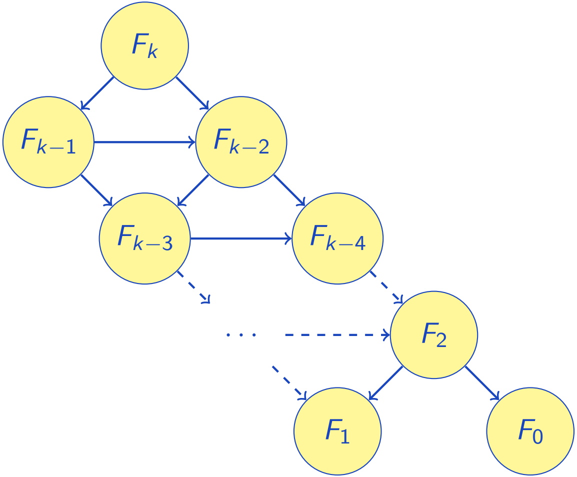
Computation of Fibonacci numbers as a graph



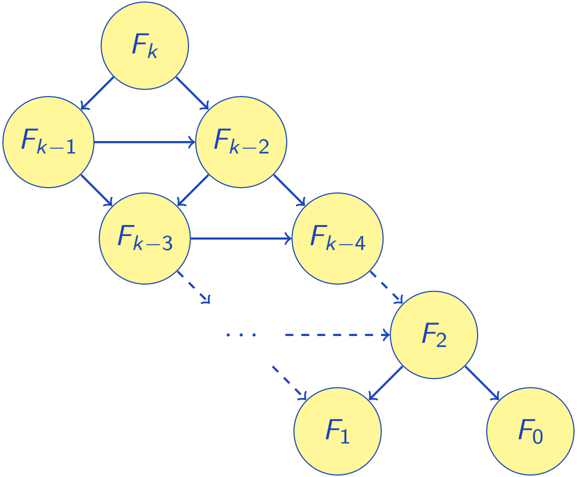
Computation of Fibonacci numbers as a graph



Computation of Fibonacci numbers as a graph

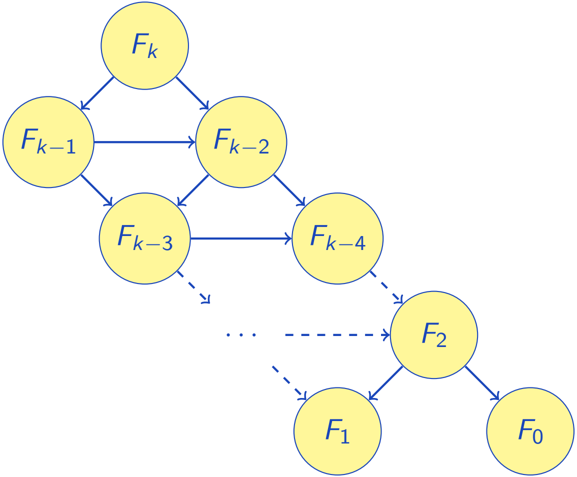


Computation of Fibonacci numbers as a graph



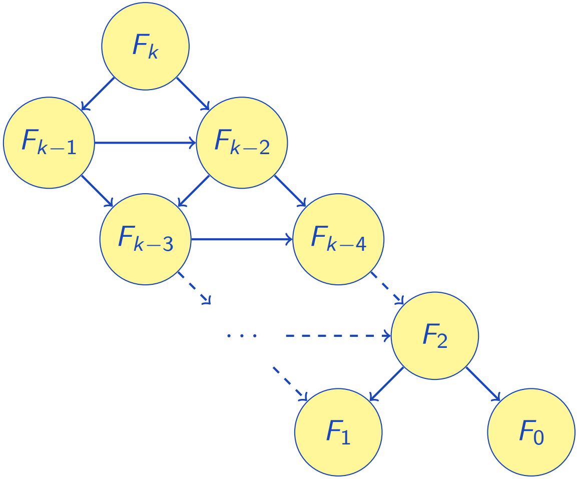
- The graph is acyclic

Computation of Fibonacci numbers as a graph



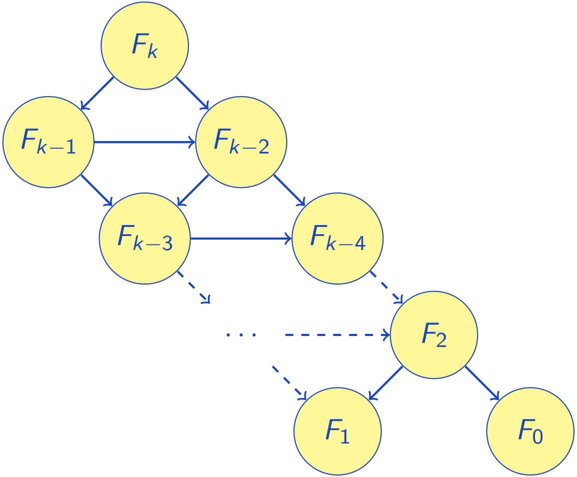
- The graph is acyclic
- Topological sort is possible

Computation of Fibonacci numbers as a graph



- The graph is acyclic
- Topological sort is possible
- Possible to evaluate each node once in the reversed order of topological sort

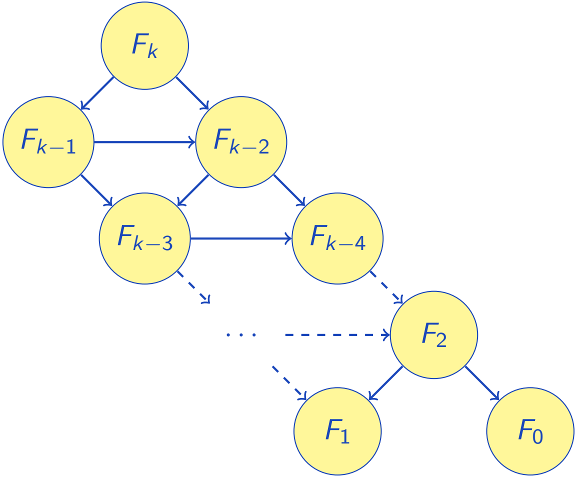
Computation of Fibonacci numbers as a graph



- The graph is acyclic
- Topological sort is possible
- Possible to evaluate each node once in the reversed order of topological sort

Dynamic programming

Computation of Fibonacci numbers as a graph



- The graph is acyclic
- Topological sort is possible
- Possible to evaluate each node once in the reversed order of topological sort

Dynamic programming

- Solves this problem in $\Theta(k)$

No graph is typically needed

- If you know there are no cycles, just store the evaluated values

Implicit topological sort

No graph is typically needed

- If you know there are no cycles, just store the evaluated values

```
int cache[MAX_N]; // add storage for the values

int fib(int k) {
    if (cache[k] != 0) { // check if the value has been computed
        return cache[k];
    }
    return cache[k] = k <= 1 // compute and store the value
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

No graph is typically needed

- If you know there are no cycles, just store the evaluated values
- Memoization

```
int cache[MAX_N]; // add storage for the values

int fib(int k) {
    if (cache[k] != 0) { // check if the value has been computed
        return cache[k];
    }
    return cache[k] = k <= 1 // compute and store the value
        ? 1
        : fib(k - 1) + fib(k - 2);
}
```

No recursion is often needed

- You can use one of possible topological sort orders if you know it
- The Fibonacci example: go from 0 to k
- The “top-down” dynamic programming
    - Compute values by running a function on dependencies

No recursion is often needed

- You can use one of possible topological sort orders if you know it
- The Fibonacci example: go from 0 to k
- The "top-down" dynamic programming
  - Compute values by running a function on dependencies

```
int fib(int k) {
    int values[k + 1]; // the storage for the values
    values[0] = values[1] = 1; // initial values
    for (int i = 2; i <= k; ++i) {
        values[i] = values[i - 1] + values[i - 2]; // top-down
    }
    return values[k];
}
```

No recursion is often needed

- You can use one of possible topological sort orders if you know it
- The Fibonacci example: go from 0 to k
- The “bottom-up” dynamic programming
    - When a value is computed, update values depending on it

No recursion is often needed

- You can use one of possible topological sort orders if you know it
- The Fibonacci example: go from 0 to k
- The "bottom-up" dynamic programming
    - When a value is computed, update values depending on it

```
int fib(int k) {
    int values[k + 2]; // the storage for the values, initially zeros
    for (int i = 0; i < k; ++i) {
        if (i <= 1) {
            values[i] = 1;
        }
        values[i + 1] += values[i]; // update one dependency
        values[i + 2] += values[i]; // update another dependency
    }
    return values[k];
}
```

Condition 1: Optimal substructure

- An (optimal) solution of the problem can be efficiently constructed from (optimal) solutions of its subproblems

Condition 1: Optimal substructure

- An (optimal) solution of the problem can be efficiently constructed from (optimal) solutions of its subproblems
- Subproblems are:
  - smaller instances of the original problem
  - smaller instances of generalized versions of the original problem

When to use dynamic programming?

Condition 1: Optimal substructure

- An (optimal) solution of the problem can be efficiently constructed from (optimal) solutions of its subproblems
- Subproblems are:
  - smaller instances of the original problem
  - smaller instances of generalized versions of the original problem

Condition 2: Overlapping subproblems

- If subproblems need totally different things to be solved,
  no need to store their solutions anywhere:
  this is a divide-and-conquer algorithm

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
S	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
?	?	X	?	?	?	?
1	?	?	?	?	?	?

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
?	?	?	?	X	?	?
1	?	X	?	?	?	?
1	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

?	X	?	?	?	?	F
?	?	?	?	X	?	?
1	?	?	?	X	?	?
1	?	X	?	?	?	?
1	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

?	X	?	?	?	?	F
1	?	?	?	X	?	?
1	?	?	?	X	?	?
1	?	X	?	?	?	?
1	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	X	?	?	?	?	F
1	?	?	?	X	?	?
1	?	?	?	X	?	?
1	?	X	?	?	?	?
1	?	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	X	?	?	?	?	F
1	?	?	?	X	?	?
1	?	?	?	X	?	?
1	?	X	?	?	?	?
1	1	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	X	?	?	?	?	F
1	?	?	?	X	?	?
1	?	?	?	X	?	?
1	2	X	?	?	?	?
1	1	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	X	?	?	?	?	F
1	?	?	?	X	?	?
1	3	?	?	X	?	?
1	2	X	?	?	?	?
1	1	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	X	?	?	?	?	F
1	4	?	?	X	?	?
1	3	?	?	X	?	?
1	2	X	?	?	?	?
1	1	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	?	?	?	?	F
1	4	?	?	X	?	?
1	3	?	?	X	?	?
1	2	X	?	?	?	?
1	1	?	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	?	?	?	?	F
1	4	?	?	X	?	?
1	3	?	?	X	?	?
1	2	X	?	?	?	?
1	1	1	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	?	?	?	?	F
1	4	?	?	X	?	?
1	3	?	?	X	?	?
1	2	0	?	?	?	?
1	1	1	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	?	?	?	?	F
1	4	?	?	X	?	?
1	3	3	?	X	?	?
1	2	0	?	?	?	?
1	1	1	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	?	?	?	?	F
1	4	7	?	X	?	?
1	3	3	?	X	?	?
1	2	0	?	?	?	?
1	1	1	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	?	?	?	F
1	4	7	?	X	?	?
1	3	3	?	X	?	?
1	2	0	?	?	?	?
1	1	1	?	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	?	?	?	F
1	4	7	?	X	?	?
1	3	3	?	X	?	?
1	2	0	?	?	?	?
1	1	1	1	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	?	?	?	F
1	4	7	?	X	?	?
1	3	3	?	X	?	?
1	2	0	1	?	?	?
1	1	1	1	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	?	?	?	`F`
1	4	7	?	X	?	?
1	3	3	4	X	?	?
1	2	0	1	?	?	?
1	1	1	1	?	?	?

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	?	?	?	F
1	4	7	11	X	?	?
1	3	3	4	X	?	?
1	2	0	1	?	?	?
1	1	1	1	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	?	?	F
1	4	7	11	X	?	?
1	3	3	4	X	?	?
1	2	0	1	?	?	?
1	1	1	1	?	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	?	?	F
1	4	7	11	X	?	?
1	3	3	4	X	?	?
1	2	0	1	?	?	?
1	1	1	1	1	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	?	?	F
1	4	7	11	X	?	?
1	3	3	4	X	?	?
1	2	0	1	2	?	?
1	1	1	1	1	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	?	?	F
1	4	7	11	X	?	?
1	3	3	4	0	?	?
1	2	0	1	2	?	?
1	1	1	1	1	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	?	?	F
1	4	7	11	0	?	?
1	3	3	4	0	?	?
1	2	0	1	2	?	?
1	1	1	1	1	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	?	F
1	4	7	11	0	?	?
1	3	3	4	0	?	?
1	2	0	1	2	?	?
1	1	1	1	1	?	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	?	F
1	4	7	11	0	?	?
1	3	3	4	0	?	?
1	2	0	1	2	?	?
1	1	1	1	1	1	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	?	F
1	4	7	11	0	?	?
1	3	3	4	0	?	?
1	2	0	1	2	3	?
1	1	1	1	1	1	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	?	F
1	4	7	11	0	?	?
1	3	3	4	0	3	?
1	2	0	1	2	3	?
1	1	1	1	1	1	?

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	?	F
1	4	7	11	0	3	?
1	3	3	4	0	3	?
1	2	0	1	2	3	?
1	1	1	1	1	1	?

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x - 1, y) + N(x, y - 1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	F
1	4	7	11	0	3	?
1	3	3	4	0	3	?
1	2	0	1	2	3	?
1	1	1	1	1	1	?

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	F
1	4	7	11	0	3	?
1	3	3	4	0	3	?
1	2	0	1	2	3	?
1	1	1	1	1	1	1

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	F
1	4	7	11	0	3	?
1	3	3	4	0	3	?
1	2	0	1	2	3	4
1	1	1	1	1	1	1

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	F
1	4	7	11	0	3	?
1	3	3	4	0	3	7
1	2	0	1	2	3	4
1	1	1	1	1	1	1

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	F
1	4	7	11	0	3	10
1	3	3	4	0	3	7
1	2	0	1	2	3	4
1	1	1	1	1	1	1

Count the number of ways to go from S to F

- using only up-moves and right-moves
- not entering a cell with X in it

Solve by dynamic programming

- Solution for a cell with S is 1
- Solution for a cell with X is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top

Example: path counting

1	0	7	18	18	21	31
1	4	7	11	0	3	10
1	3	3	4	0	3	7
1	2	0	1	2	3	4
1	1	1	1	1	1	1

Count the number of ways to go from `S` to `F`

- using only up-moves and right-moves
- not entering a cell with `X` in it

Solve by dynamic programming

- Solution for a cell with `S` is 1
- Solution for a cell with `X` is 0
- Solution for a cell $N(x, y)$ is $N(x-1, y) + N(x, y-1)$
- Possible traversal order: left to right, bottom to top