Deterministic Finite Automata

Arnab Chakraborty

Indian Statistical Institute

Sep 15, 2023

Don't worry...

...this is not a G-pay QR code:

...just a link to a Youtube playlist containing this lecture.

Deterministic Finite Automaton (DFA)

= Finite Automaton

= Finite state machine

= State machine

= Internal state machine

Application areas

Computer science:

- **Robotics**
- **Digital electronics**
- **Compilers for languages like C, Java etc**
- **Embedded systems (*e.g.*, IoT)**

Mathematics:

1. **Algebraic structures**
2. **Combinatorics**

Humptydumpty

7+3=10

Definition

Definition

Three sets:

- A = **set of inputs** = $\{$**Smile, Punch**$\}$
- B = **set of outputs** = $\{$**Normal, Smile back, Raise eyebrow, Scowl, Growl, Grimace**$\}$
- S = **set of states** = $\{$**Ordinary, Happy, Surprised, Angry, Furious**$\}$

Definition

Three sets:

- $A =$ **set of inputs =** $\{$**Smile, Punch**$\}$
- $B =$ **set of outputs =** $\{$**Normal, Smile back, Raise eyebrow, Scowl, Growl, Grimace**$\}$
- $S =$ **set of states =** $\{$**Ordinary, Happy, Surprised, Angry, Furious**$\}$

Two functions:

- $f : A \times S \to B$, **the output function.**
- $g : A \times S \to S$, **the next state function.**

Definition

Three sets:

- A = **set of inputs =** {Smile, Punch}
- B = **set of outputs =** {Normal, Smile back, Raise eyebrow, Scowl, Growl, Grimace}
- S = **set of states =** {Ordinary, Happy, Surprised, Angry, Furious}

Two functions:

- $f : A \times S \to B$, **the output function.**
- $g : A \times S \to S$, **the next state function.**

One state:

- $s_0 \in S$: **the initial state = Ordinary.**

10+3=13

Transition diagram

Why care?

Why care?

Any machine with a memory must be a state machine!

C comments


```
void main() {
  printf("Hello world!"); /* Correct this.*/
}
```

16+1=17

Look for /* ... */

16+1=17

Look for /* ... */

Look for /* ... */

Look for /* ... */

x=4

Look for /* ... */

x=4/

Look for /* ... */

x=4/5

Look for /* ... */

x=4/5+

Look for /* ... */

x=4/5+3

16+1=17

Look for /* ... */

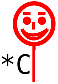
x=4/5+3/

Look for /* ... */


```
x=4/5+3/*
```

Look for /* ... */

x=4/5+3/*C



Look for /* ... */

```
x=4/5+3/*Ch
```

Look for /* ... */


```
x=4/5+3/*Cha
```

Look for /* ... */

```
x=4/5+3/*Chan
```

Look for /* ... */

```
x=4/5+3/*Chang
```

Look for /* ... */


```
x=4/5+3/*Change
```

Look for /* ... */


```
x=4/5+3/*Change
```

Look for /* ... */


```
x=4/5+3/*Change /
```

Look for /* ... */


```
x=4/5+3/*Change /
```

16+1=17

Look for /* ... */

```
x=4/5+3/*Change / t
```

Look for /* ... */


```
x=4/5+3/*Change / to
```

Look for /* ... */

```
x=4/5+3/*Change / to
```

Look for /* ... */


```
x=4/5+3/*Change / to *
```

Look for /* ... */

```
x=4/5+3/*Change / to *.
```

Look for /* ... */


```
x=4/5+3/*Change / to *.*
```

Look for /* ... */


```
x=4/5+3/*Change / to *.*/
```

Look for /* ... */


```
x=4/5+3/*Change / to *.*/;
```

17+3=20

Inside our brain

Looking for start of comment.

17+3=20

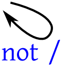
Inside our brain

Looking for start of comment.

not /

Inside our brain

Looking for start of comment.



17+3=20

Inside our brain

x=4

Looking for start of comment.

not /

Inside our brain

x=4/ 

Looking for start of comment.

/

not /

Possible start.

Inside our brain

x=4/5

Looking for start of comment.

not *

/

not /

Possible start.

Inside our brain

x=4/5+

Inside our brain

x=4/5+3

Inside our brain

x=4/5+3/

Looking for start of comment.

not *

/

not /

Possible start.

Inside our brain

x=4/5+3/*

Looking for start of comment.

not /

not *

/

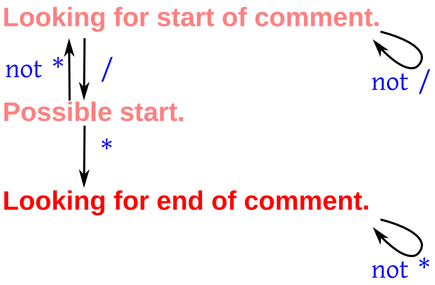
Possible start.

*

Looking for end of comment.

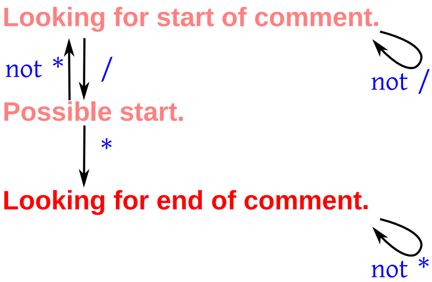
Inside our brain


```
x=4/5+3/*C
```



Inside our brain

```
x=4/5+3/*Ch
```



Inside our brain


```
x=4/5+3/*Cha
```

Looking for start of comment.

not /

not * | /

Possible start.

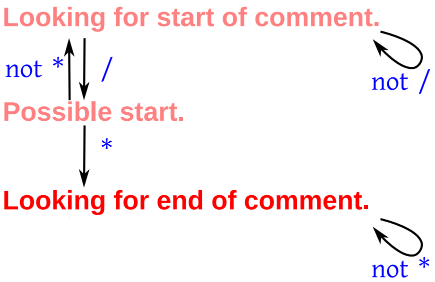
*

Looking for end of comment.

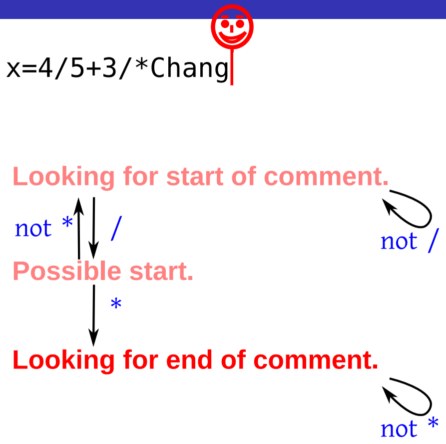
not *

Inside our brain

```
x=4/5+3/*Chan
```

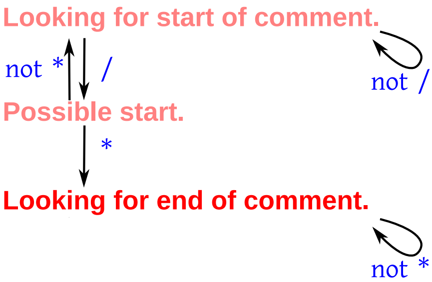


Inside our brain



Inside our brain

```
x=4/5+3/*Change
```



Inside our brain

```
x=4/5+3/*Change
```

Looking for start of comment.

not * /

not /

Possible start.

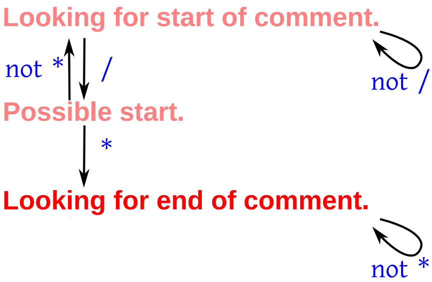
*

Looking for end of comment.

not *

Inside our brain

```
x=4/5+3/*Change /
```



Inside our brain

```
x=4/5+3/*Change /
```

Looking for start of comment.

not /

not * / /

Possible start.

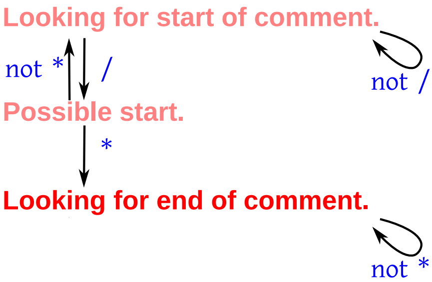
*

Looking for end of comment.

not *

Inside our brain

`x=4/5+3/*Change / t`



Inside our brain


```
x=4/5+3/*Change / to
```

Looking for start of comment.

not /

not * /

Possible start.

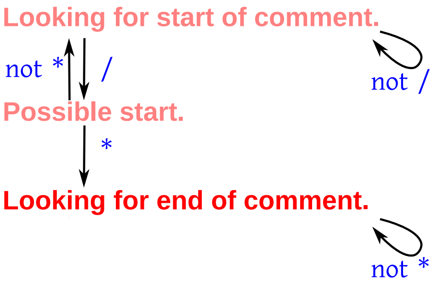
*

Looking for end of comment.

not *

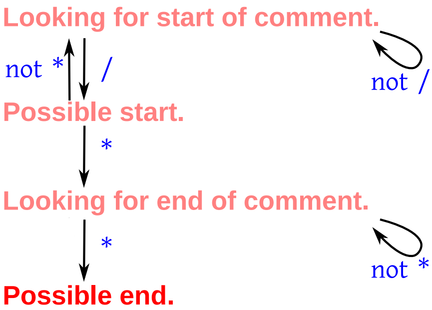
Inside our brain

```
x=4/5+3/*Change / to
```



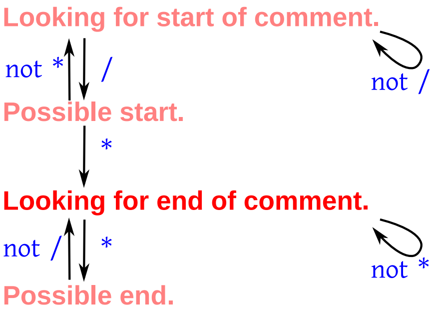
Inside our brain


```
x=4/5+3/*Change / to *
```



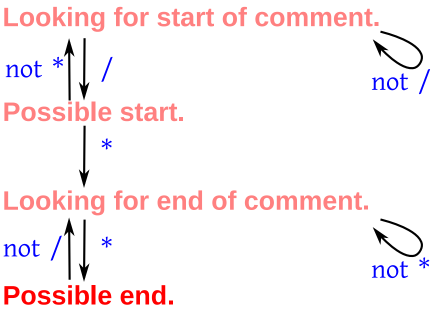
Inside our brain

```
x=4/5+3/*Change / to *.
```



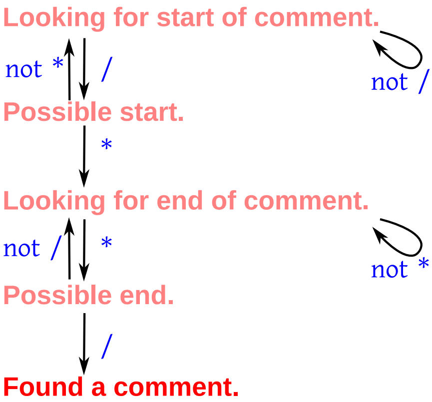
Inside our brain

```
x=4/5+3/*Change / to *.*
```



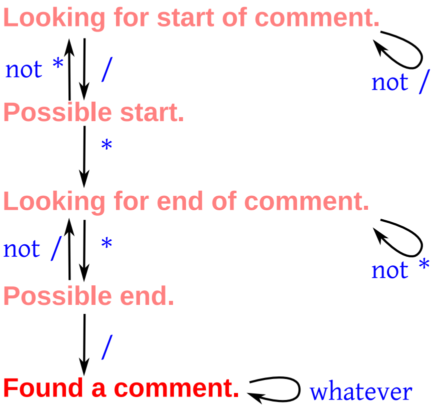
Inside our brain


```
x=4/5+3/*Change / to *.*/
```



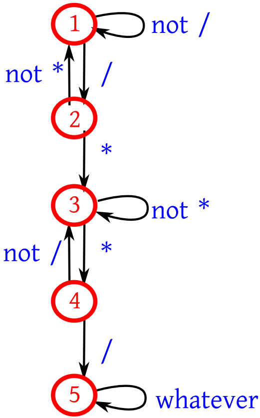
Inside our brain

```
x=4/5+3/*Change / to *.*/;
```



Inside our brain

```
x=4/5+3/*Change / to *.*/;
```



$20+2=22$

Demo

English to Bengali transliteration

Application in hardware

Application in hardware

Arduino

24+2=26

Application in hardware

Arduino

Raspberry Pi

Coin dropper example

Coin dropper example

Coin dropper example

Mathematically...

...a state machine is much like a ...

Mathematically...

...a state machine is much like a monoid.

Mathematically...

...a state machine is much like a monoid.

A monoid is $(M, +)$ where M is a set and $+$ is an associative binary operation on M having identity.

Mathematically...

...a state machine is much like a monoid.

A monoid is $(M, +)$ where M is a set and $+$ is an associative binary operation on M having identity.

Example:

$(\{0, 1, 2, ...\}, +)$.

Mathematically...

...a state machine is much like a monoid.

A monoid is $(M, +)$ where M is a set and $+$ is an associative binary operation on M having identity.

Example:

$(\{0, 1, 2, ...\}, +)$. **Identity = 0.**

Mathematically...

...a state machine is much like a monoid.

A monoid is $(M, +)$ where M is a set and $+$ is an associative binary operation on M having identity.

Example:

All finite length words using the letters a, b, c under the concatenation operator: $"abc" \bullet "ab" = "abcab"$.

Mathematically...

...a state machine is much like a monoid.

A monoid is $(M, +)$ where M is a set and $+$ is an associative binary operation on M having identity.

Example:

All finite length words using the letters a, b, c under the concatenation operator: $"abc" \bullet "ab" = "abcab"$.
Identity = "".

From finite state machines to monoids

From finite state machines to monoids

Certain input words are equivalent for the machine.

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

$$w_3 = w_1 \bullet w_2 .$$

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

$$[w_3] = [w_1] \bullet [w_2].$$

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

$$[w_3] = [w_1] \bullet [w_2].$$

This defines a new operation on the equivalence classes, under which they form a monoid.

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

$$[w_3] = [w_1] \bullet [w_2].$$

This defines a new operation on the equivalence classes, under which they form a monoid. This monoid captures the "underlying structure" of the finite machine.

From finite state machines to monoids

Certain input words are equivalent for the machine.

Club them together. You will get finitely many classes.

$$[w_3] = [w_1] \bullet [w_2].$$

This defines a new operation on the equivalence classes, under which they form a monoid. This monoid captures the "underlying structure" of the finite machine.

The monoid is always finite.

From monoid to finite state machine

Start with a finite monoid, $(M, +)$. Then we can define a FSM with state set and input set both equal to M. For any state $s \in M$ and any input $i \in M$ we define the next state as $m + i$.

Some interesting questions

Some interesting questions

- **Monoid to FSM to monoid.**

Some interesting questions

- **Monoid to FSM to monoid.**
- **FSM to monoid to FSM.**

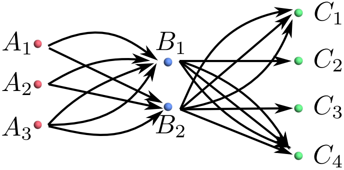
Some interesting questions

- **Monoid to FSM to monoid.**
- **FSM to monoid to FSM.**
- **Size of the monoid for a given FSM.**

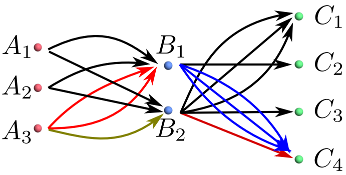
Some interesting questions

- **Monoid to FSM to monoid.**
- **FSM to monoid to FSM.**
- **Size of the monoid for a given FSM.**
- **Same monoid $\implies$ same transition diagram?**

Transfer matrix method

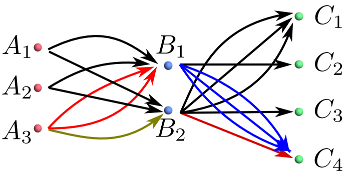


Transfer matrix method



$$2 \times 3 + 1 \times 1 = 7.$$

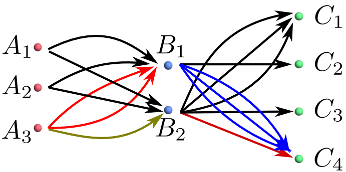
Transfer matrix method



$$\color{red}{2} \times \color{blue}{3} + \color{green}{1} \times \color{purple}{1} = 7.$$

$$\begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \end{array} \begin{array}{c} \begin{array}{cc} B_1 & B_2 \end{array} \\ \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ \mathbf{2} & \mathbf{1} \end{bmatrix} \end{array} \qquad \begin{array}{c} \\ B_1 \\ B_2 \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & \mathbf{3} \\ 3 & 0 & 1 & \mathbf{1} \end{bmatrix} \end{array}$$

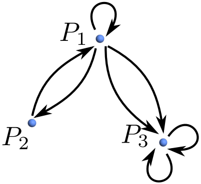
Transfer matrix method



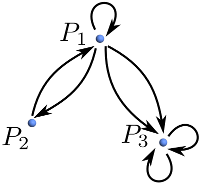
$$2 \times 3 + 1 \times 1 = 7.$$

$$\begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \end{array}\begin{array}{c} \begin{array}{cc} B_1 & B_2 \end{array} \\ \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ \mathbf{2} & \mathbf{1} \end{bmatrix} \end{array} \times \begin{array}{c} \\ B_1 \\ B_2 \end{array}\begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & \mathbf{3} \\ 3 & 0 & 1 & \mathbf{1} \end{bmatrix} \end{array} = \begin{array}{c} \\ A_1 \\ A_2 \\ A_3 \end{array}\begin{array}{c} \begin{array}{cccc} C_1 & C_2 & C_3 & C_4 \end{array} \\ \begin{bmatrix} 3 & 1 & 1 & 4 \\ 3 & 1 & 1 & 4 \\ 3 & 2 & 1 & \mathbf{7} \end{bmatrix} \end{array}$$

Same set

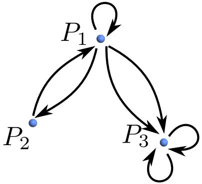


Same set



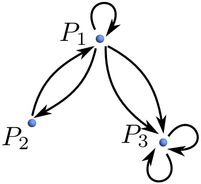
$$\begin{array}{c c} & \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{array}{c} P_1 \\ P_2 \\ P_3 \end{array} & \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \end{array}$$

Same set



$$\begin{array}{cc} & \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{array}{c} P_1 \\ P_2 \\ P_3 \end{array} & \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \end{array} = A, \text{ say.}$$

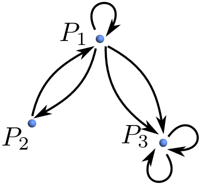
Same set



$$\begin{array}{c c} & \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{array}{c} P_1 \\ P_2 \\ P_3 \end{array} & \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \end{array} = A, \text{ say.}$$

$$A^2 = \begin{bmatrix} 2 & 1 & 6 \\ 1 & 1 & 2 \\ 0 & 0 & 4 \end{bmatrix}.$$

Same set



$$\begin{array}{c} \\ P_1 \\ P_2 \\ P_3 \end{array} \begin{array}{c} \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{bmatrix} \end{array} = A, \text{ say.}$$

$$A^2 = \begin{bmatrix} 2 & 1 & 6 \\ 1 & 1 & 2 \\ 0 & 0 & 4 \end{bmatrix}.$$

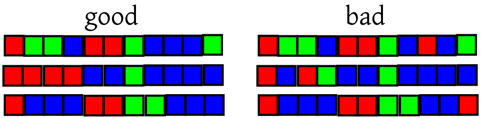
The (i, j)-th entry of A^k gives the number of paths of length k from P_i to P_j.

Example: a counting problem

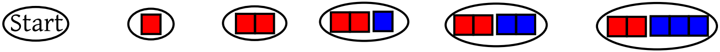
Want to count the number (b_n) of sequences with , and . Total number of squares is n. Must have two consecutive 's somewhere before three consecutive 's.

Example: a counting problem

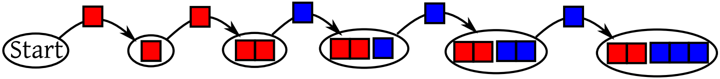
Want to count the number (b_n) of sequences with 🟥, 🟩 and 🟦. Total number of squares is n. Must have two consecutive 🟥's somewhere before three consecutive 🟦's.



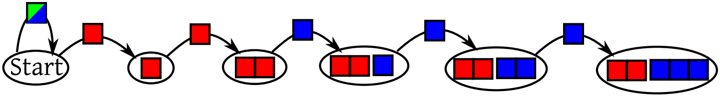
Solution



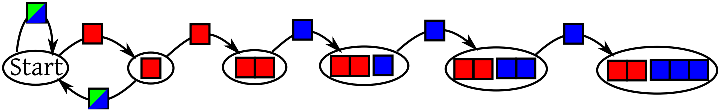
Solution



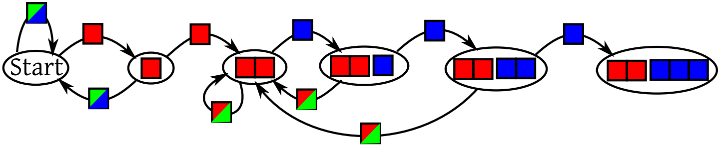
Solution



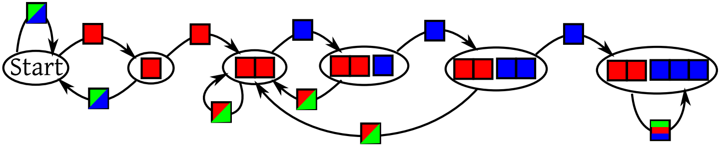
Solution



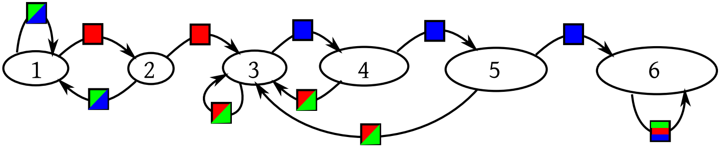
Solution



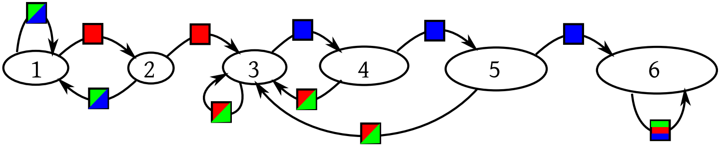
Solution



Solution

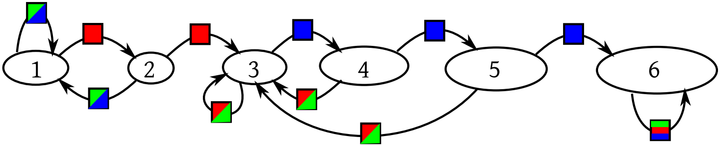


Solution



$$A = \begin{array}{c} \text{From}\backslash\text{To} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix} \end{array}$$

Solution



$$A = \begin{array}{c|c} {}_{\text{From}}\backslash^{\text{To}} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix} \end{array}$$

The $(1,6)$-th entry of A^n gives the number of such patterns of length n, *i.e.* b_n.

Example

`http://maxima.cesga.es`

Example

http://maxima.cesga.es

```
A: matrix([2, 1, 0, 0, 0, 0],
          [2, 0, 1, 0, 0, 0],
          [0, 0, 2, 1, 0, 0],
          [0, 0, 2, 0, 1, 0],
          [0, 0, 2, 0, 0, 1],
          [0, 0, 0, 0, 0, 3]);
```

Example


```
http://maxima.cesga.es

A: matrix([2, 1, 0, 0, 0, 0],
          [2, 0, 1, 0, 0, 0],
          [0, 0, 2, 1, 0, 0],
          [0, 0, 2, 0, 1, 0],
          [0, 0, 2, 0, 0, 1],
          [0, 0, 0, 0, 0, 3]);

A^^45;
```

Example


```
http://maxima.cesga.es

A: matrix([2, 1, 0, 0, 0, 0],
          [2, 0, 1, 0, 0, 0],
          [0, 0, 2, 1, 0, 0],
          [0, 0, 2, 0, 1, 0],
          [0, 0, 2, 0, 0, 1],
          [0, 0, 0, 0, 0, 3]);

A^^45;
```

$$\begin{bmatrix} 3458540538388755 2512 & 12659136970685939712 & 86357776463132858 7776 & 29467125259992591 5648 & 10052059169947344 8960 & 16482985552655322 54035 \\ 25318273941371879424 & 9267131442515673088 & 80304282556948466 8928 & 27423525943147675 6480 & 9363006920097901 7728 & 17488191469650057 02995 \\ 0 & 0 & 62392791058373938 3808 & 21370032036963283 7632 & 7319407603252985 8560 & 20434903995649316 18643 \\ 0 & 0 & 57378879280432539 2384 & 19652726984447370 8544 & 6731216830457312 0512 & 21166844755974614 77203 \\ 0 & 0 & 42740064073926567 5264 & 14638815206505971 7120 & 5013911777941399 1424 & 23303847959670943 14835 \\ 0 & 0 & 0 & 0 & 0 & 29543127065508336 98643 \end{bmatrix}$$

16482985552655322 54035

Analytically...

The generating function of $\{b_n\}_n$ is the formal power series

$$b_0 + b_1 t + b_2 t^2 + b_3 t^3 + \cdots .$$

Analytically...

The generating function of $\{b_n\}_n$ is the formal power series

$$b_0 + b_1 t + b_2 t^2 + b_3 t^3 + \cdots .$$

This is the $(1,6)$-th entry of

$$I + At + A^2 t^2 + \cdots$$

Analytically...

The generating function of $\{b_n\}_n$ is the formal power series

$$b_0 + b_1 t + b_2 t^2 + b_3 t^3 + \cdots .$$

This is the $(1,6)$-th entry of

$$I + At + A^2 t^2 + \cdots = (I - tA)^{-1}.$$

Analytically...

The generating function of $\{b_n\}_n$ is the formal power series

$$b_0 + b_1 t + b_2 t^2 + b_3 t^3 + \cdots .$$

This is the $(1,6)$-th entry of

$$I + At + A^2 t^2 + \cdots = (I - tA)^{-1}.$$

Just like

$$1 + x + x^2 + \cdots = \frac{1}{1-x} \textbf{ if } |x| < 1.$$

Example


```
(1-t*A);
```

$$\begin{bmatrix} 1-2\,t & 1-t & 1 & 1 & 1 & 1 \\ 1-2\,t & 1 & 1-t & 1 & 1 & 1 \\ 1 & 1 & 1-2\,t & 1-t & 1 & 1 \\ 1 & 1 & 1-2\,t & 1 & 1-t & 1 \\ 1 & 1 & 1-2\,t & 1 & 1 & 1-t \\ 1 & 1 & 1 & 1 & 1 & 1-3\,t \end{bmatrix}$$

Inverse

```
((I-t*A)^^-1)[1,6];
```

Inverse


```
((I-t*A)^^-1)[1,6];
```

$$\frac{t^5}{-12\,t^6 - 20\,t^5 - 10\,t^4 + 6\,t^3 + 12\,t^2 - 7\,t + 1}$$

Inverse


```
((I-t*A)^^-1)[1,6];
```

$$\frac{t^5}{-12\,t^6 - 20\,t^5 - 10\,t^4 + 6\,t^3 + 12\,t^2 - 7\,t + 1}$$

From this we get

$$b_{n+5} = 7b_{n+4} - 12b_{n+3} - 6b_{n+2} + 10b_{n+1} + 20b_n + 12b_{n-1}$$

Inverse


```
((I-t*A)^^-1)[1,6];
```

$$\frac{t^5}{-12\,t^6 - 20\,t^5 - 10\,t^4 + 6\,t^3 + 12\,t^2 - 7\,t + 1}$$

From this we get

$$b_{n+5} = 7b_{n+4} - 12b_{n+3} - 6b_{n+2} + 10b_{n+1} + 20b_n + 12b_{n-1}$$

How?

Reference