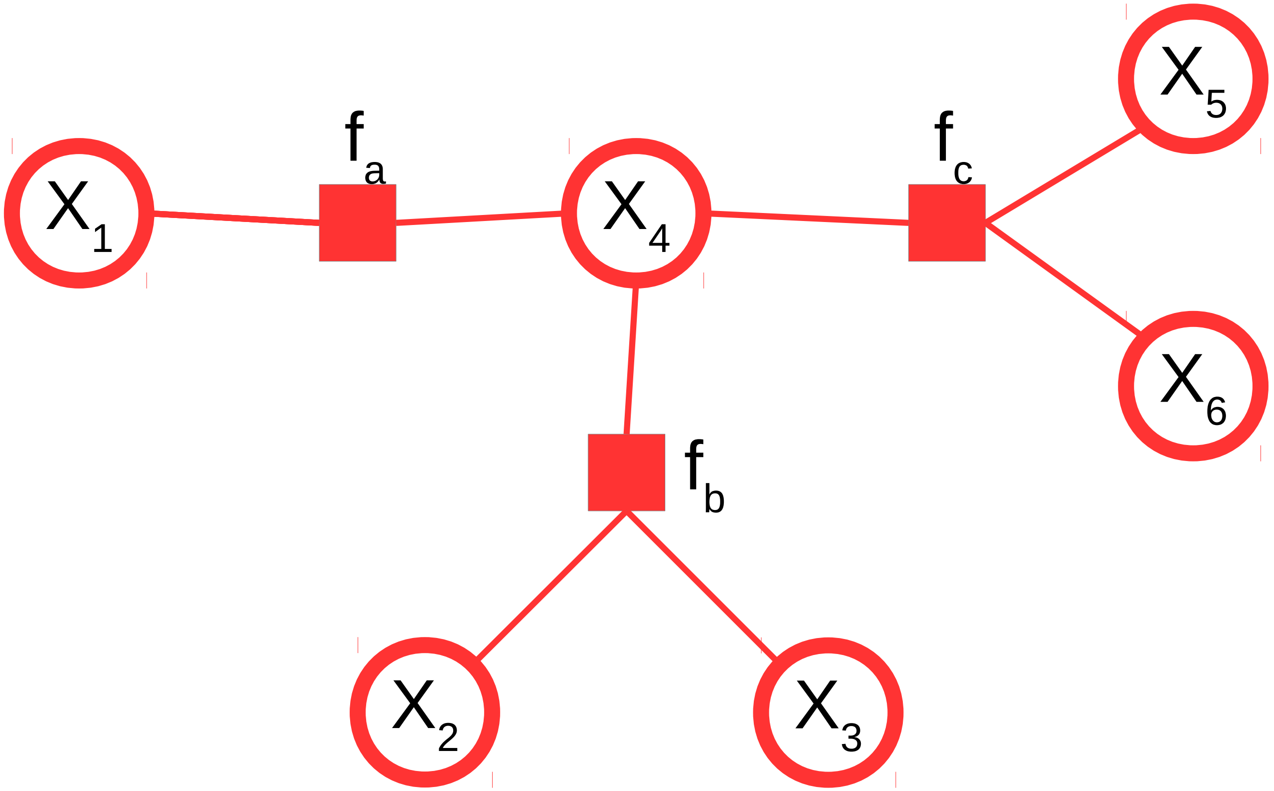
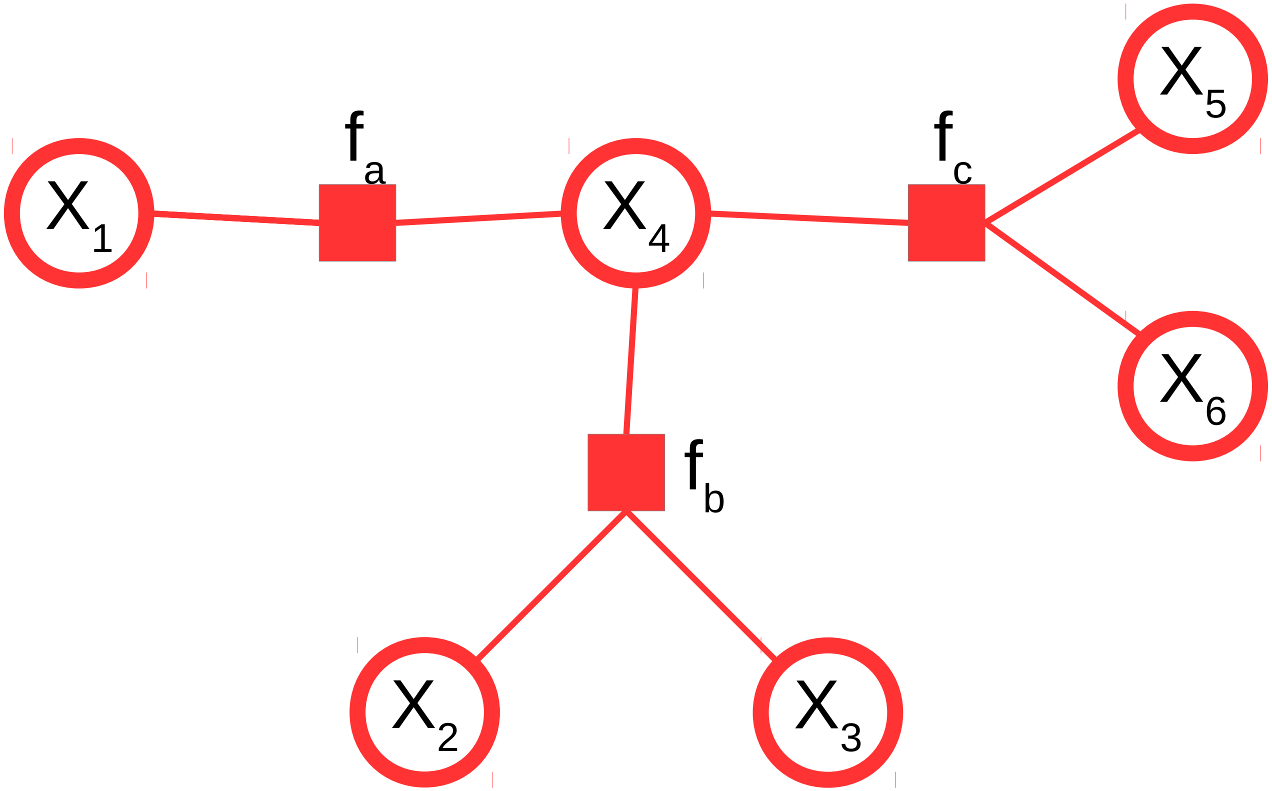
ETH zürich

Inference in factor trees

Carlos Cotrini
November 3, 2017

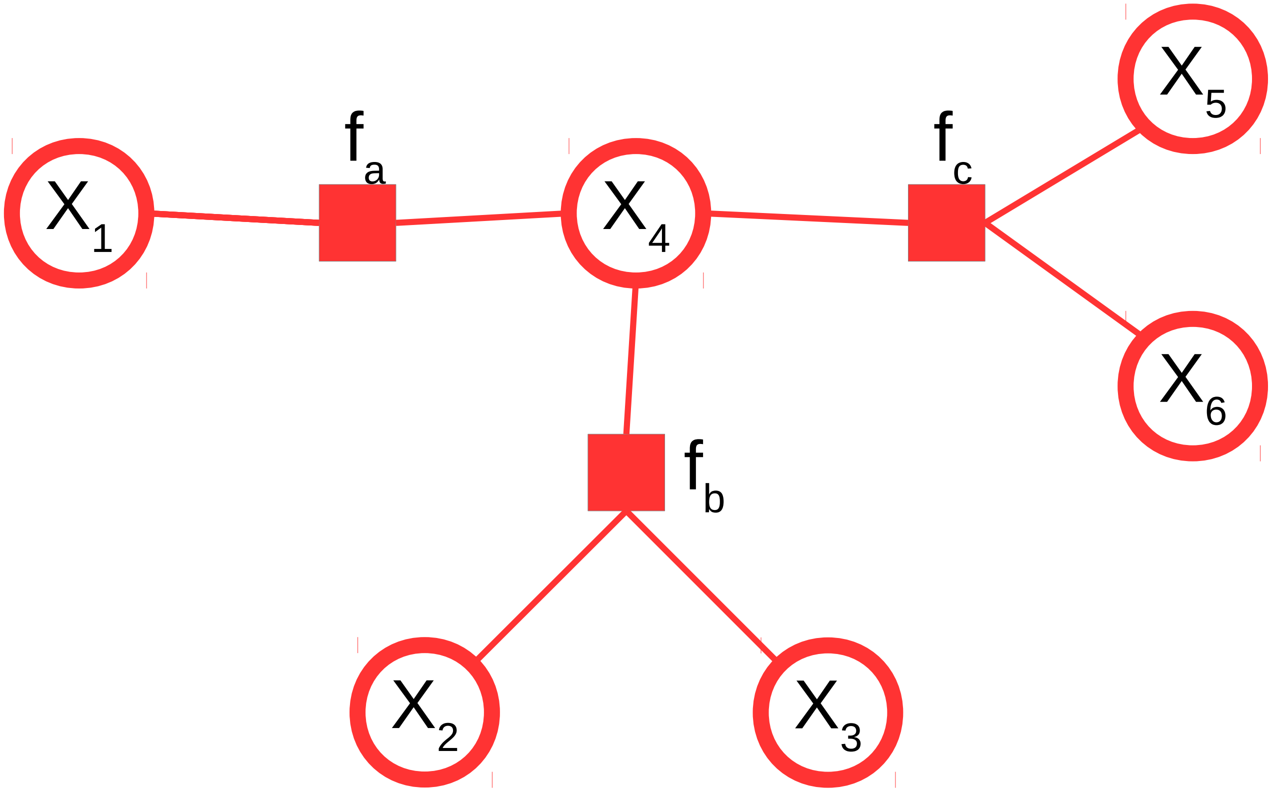
Probabilistic foundations of artificial intelligence


X_1
f_a
X_4
f_c
X_5
X_6
f_b
X_2
X_3



$$P(X_1 = \hat{x}_1, \ldots, X_6 = \hat{x}_6) =$$

$$\frac{1}{Z} f_a(\hat{x}_1, \hat{x}_4) f_b(\hat{x}_2, \hat{x}_3, \hat{x}_4) f_c(\hat{x}_4, \hat{x}_5, \hat{x}_6)$$



$$P(X_1 = \hat{x}_1, \ldots, X_6 = \hat{x}_6) =$$

$$\frac{1}{Z} f_a(\hat{x}_1, \hat{x}_4) f_b(\hat{x}_2, \hat{x}_3, \hat{x}_4) f_c(\hat{x}_4, \hat{x}_5, \hat{x}_6)$$

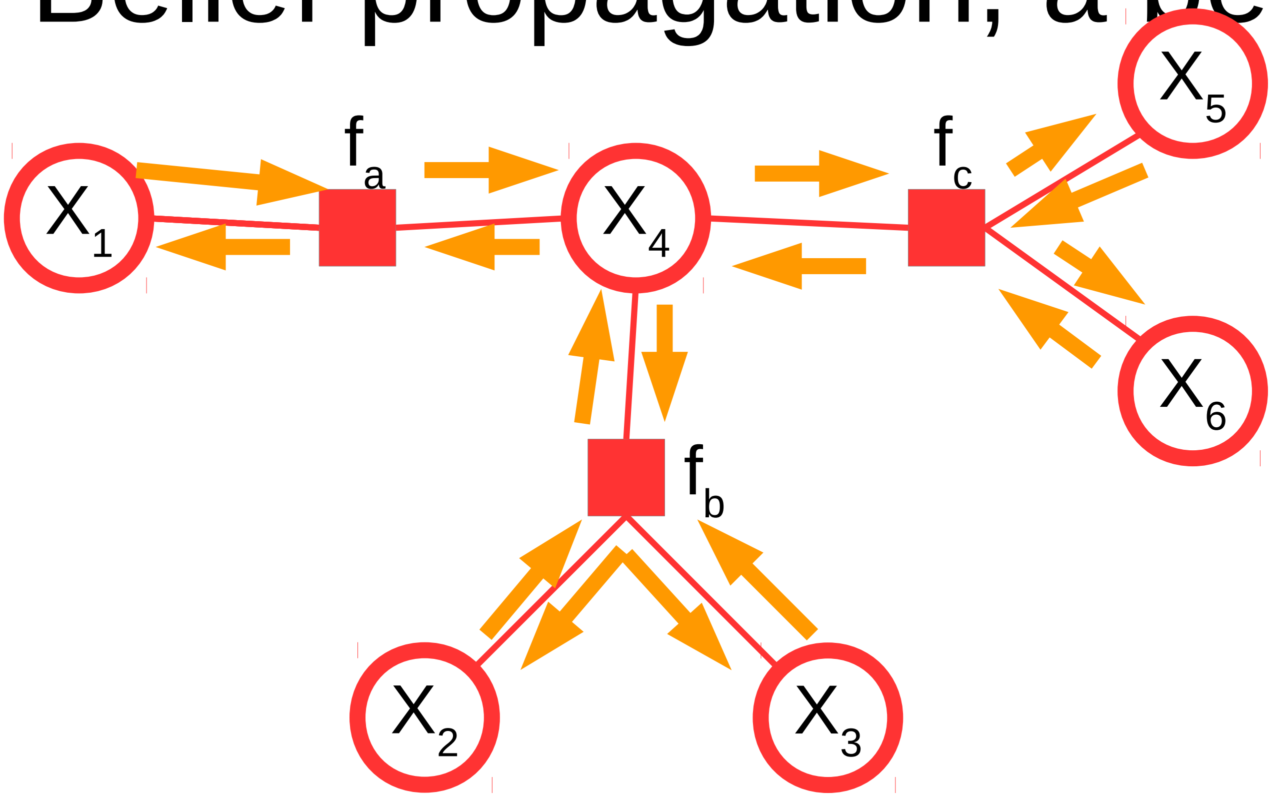
How do we compute $P(X_5 = \hat{x}_5)$?

ETH zürich

Naive method

$$P(X_5 = \hat{x}_5) = \sum_{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4, \hat{x}_6} P(X_1 = \hat{x}_1, X_2 = \hat{x}_2, X_3 = \hat{x}_3, X_4 = \hat{x}_4, X_6 = \hat{x}_6).$$

Belief propagation, a better method.



Belief propagation, a better method.

$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$

- X: a node (i.e., a random variable).
- f: a factor.
- $\hat{x}$: a value in the range of X.
- $N(X)$: X's neighbors.
- $N(f)$: f's neighbors.
- $\hat{\mathbf{x}}$: a sequence of values in f's domain.

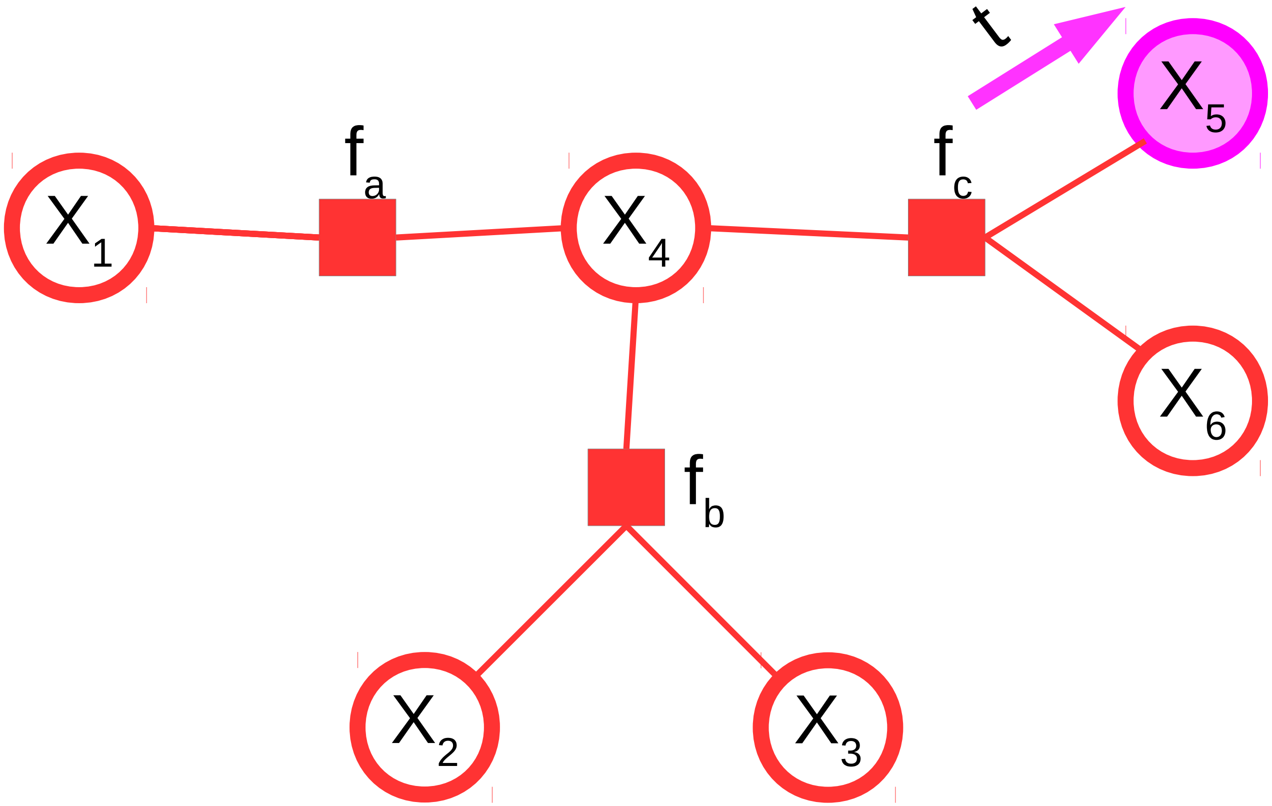
Initially,
$\mu^{(0)}_{X\to f}(\hat{x}) = 1$ and
$\mu^{(0)}_{f\to X}(\hat{x}) = 1$.

Before we compute $P(X_5 = x_5)$, let's observe three useful insights about belief propagation in trees.

First insight

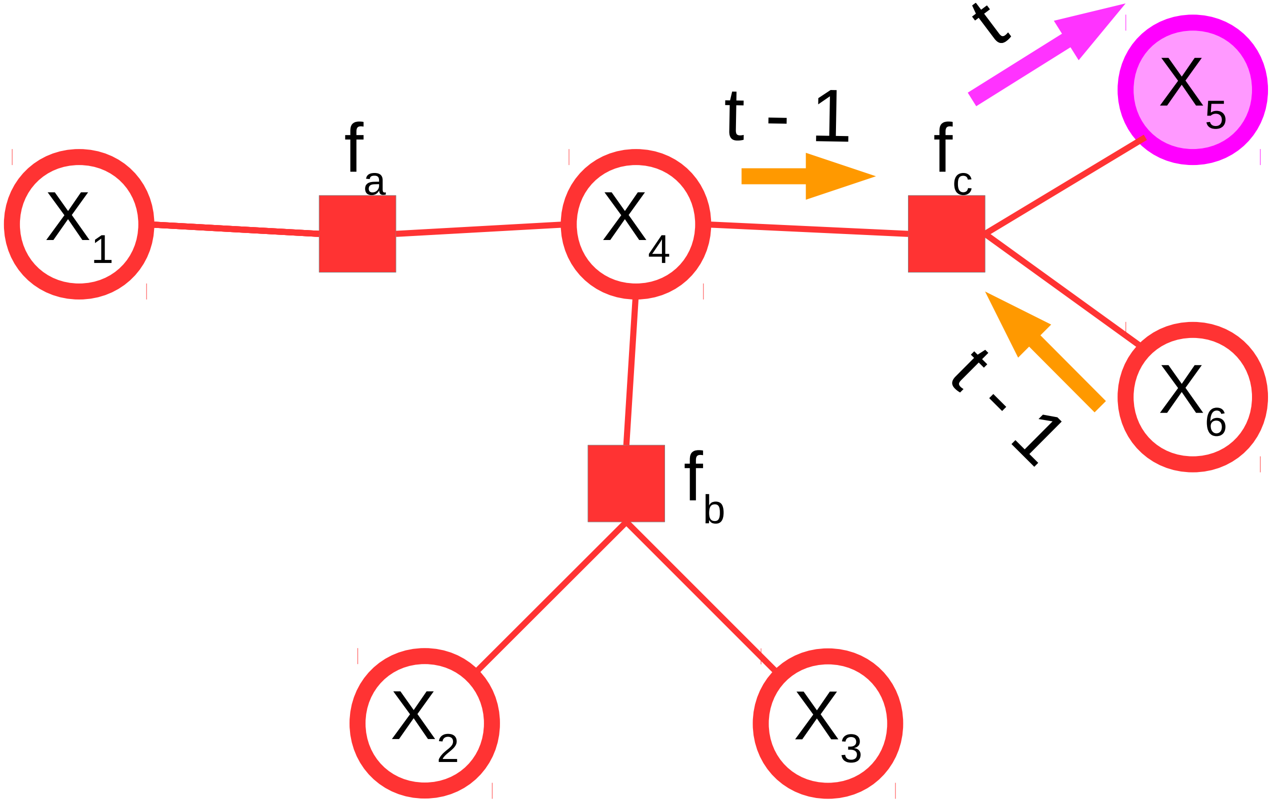
The messages needed to compute another message form a tree*

* This only holds for factor trees!



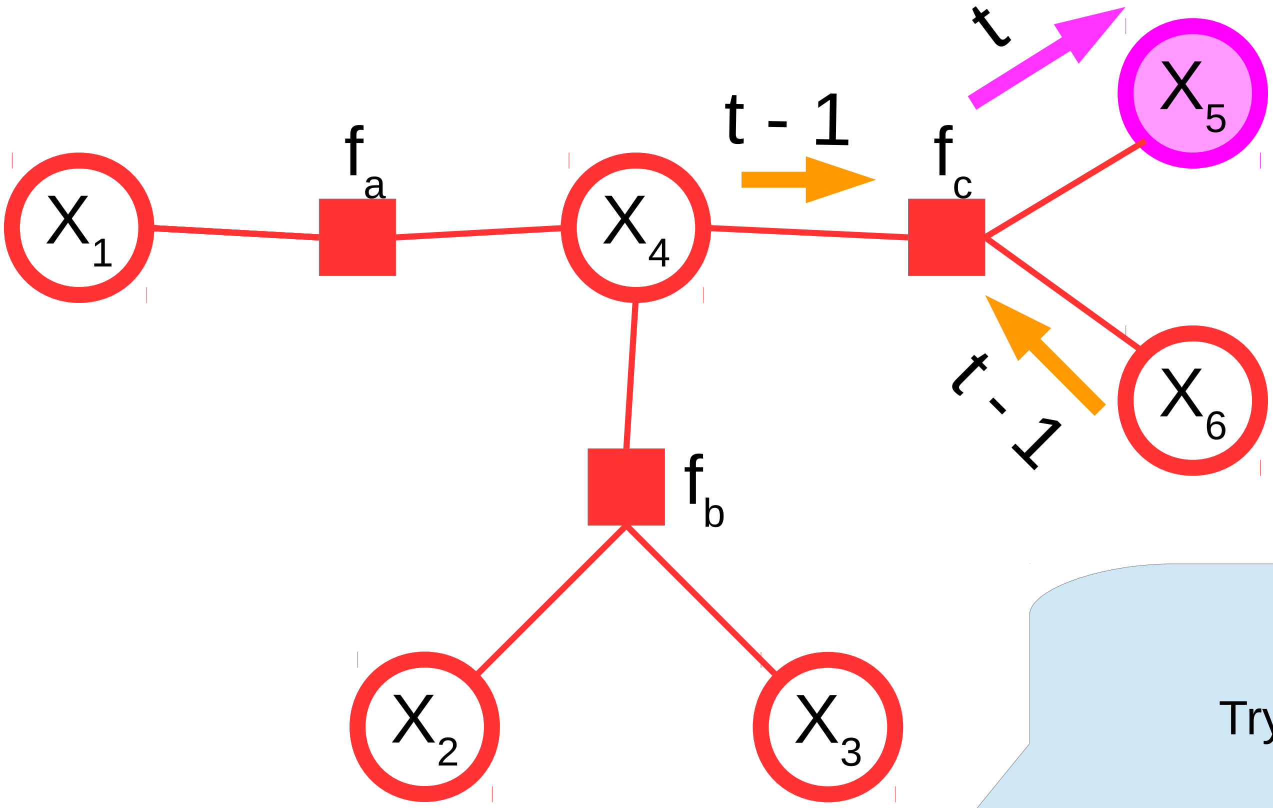
$$\mu^{(t)}_{X \to f}(\hat{x}) = \prod_{f' \in N(X) \setminus \{f\}} \mu^{(t-1)}_{f' \to X}(\hat{x})$$

$$\mu^{(t)}_{f \to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X' \in N(f) \setminus \{X\}} \mu^{(t-1)}_{X' \to f}(\hat{x}')$$



$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

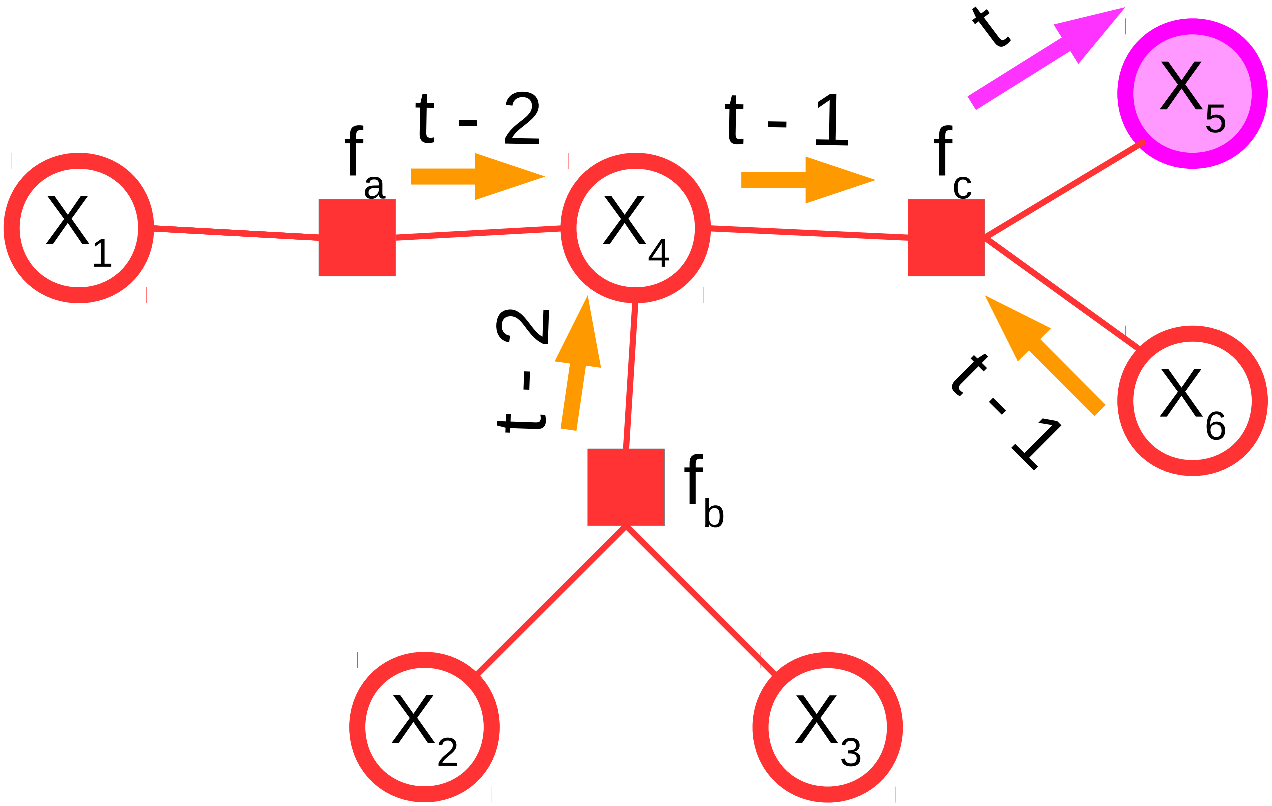
$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



Try the rest by yourself

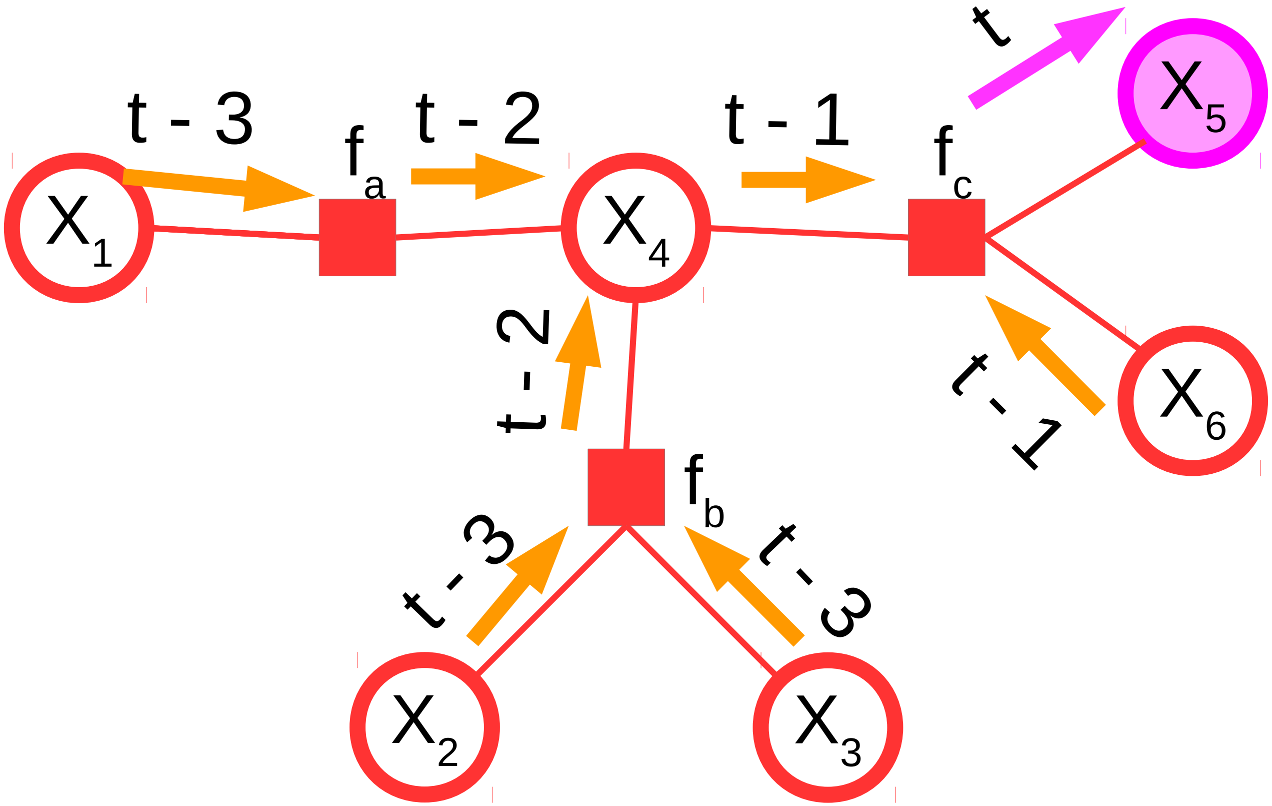
$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f' \in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X' \in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



$$\mu^{(t)}_{X \to f}(\hat{x}) = \prod_{f' \in N(X) \setminus \{f\}} \mu^{(t-1)}_{f' \to X}(\hat{x})$$

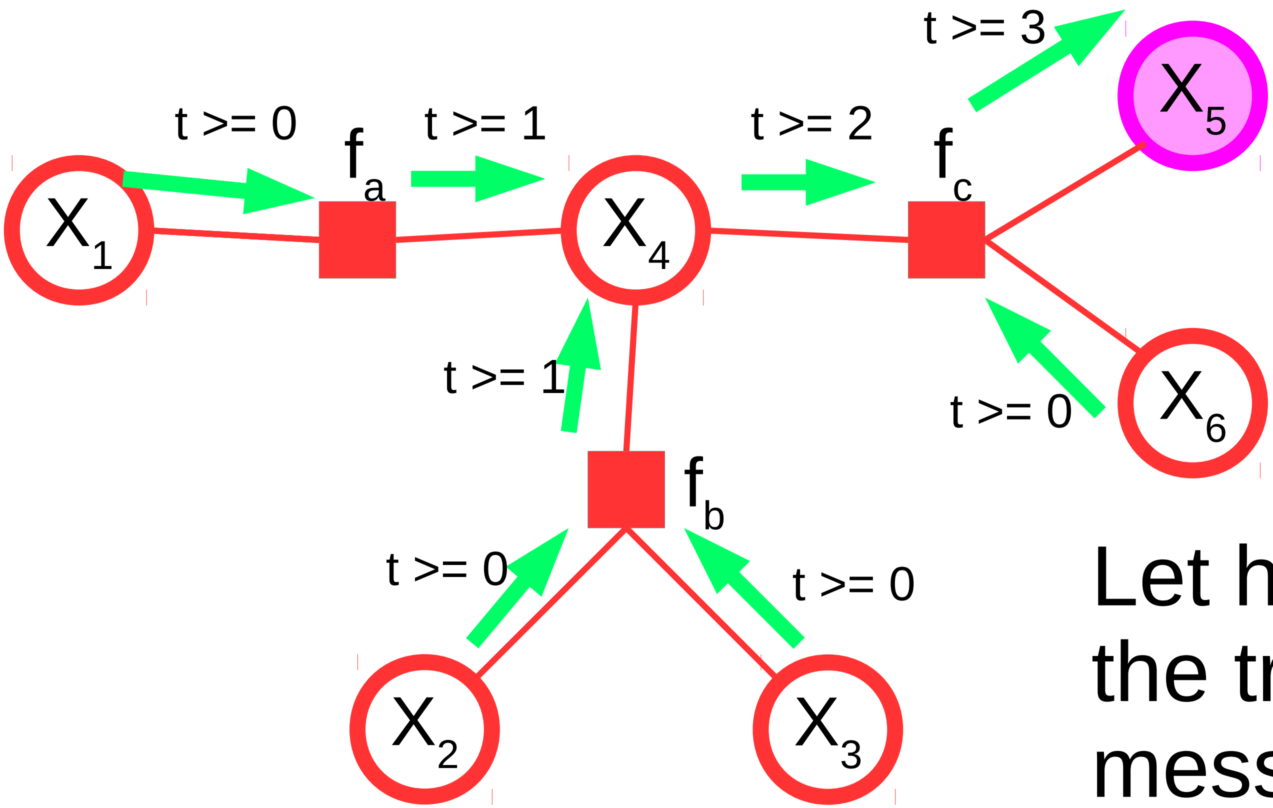
$$\mu^{(t)}_{f \to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X' \in N(f) \setminus \{X\}} \mu^{(t-1)}_{X' \to f}(\hat{x}')$$

First insight

The messages needed to compute another message form a tree*

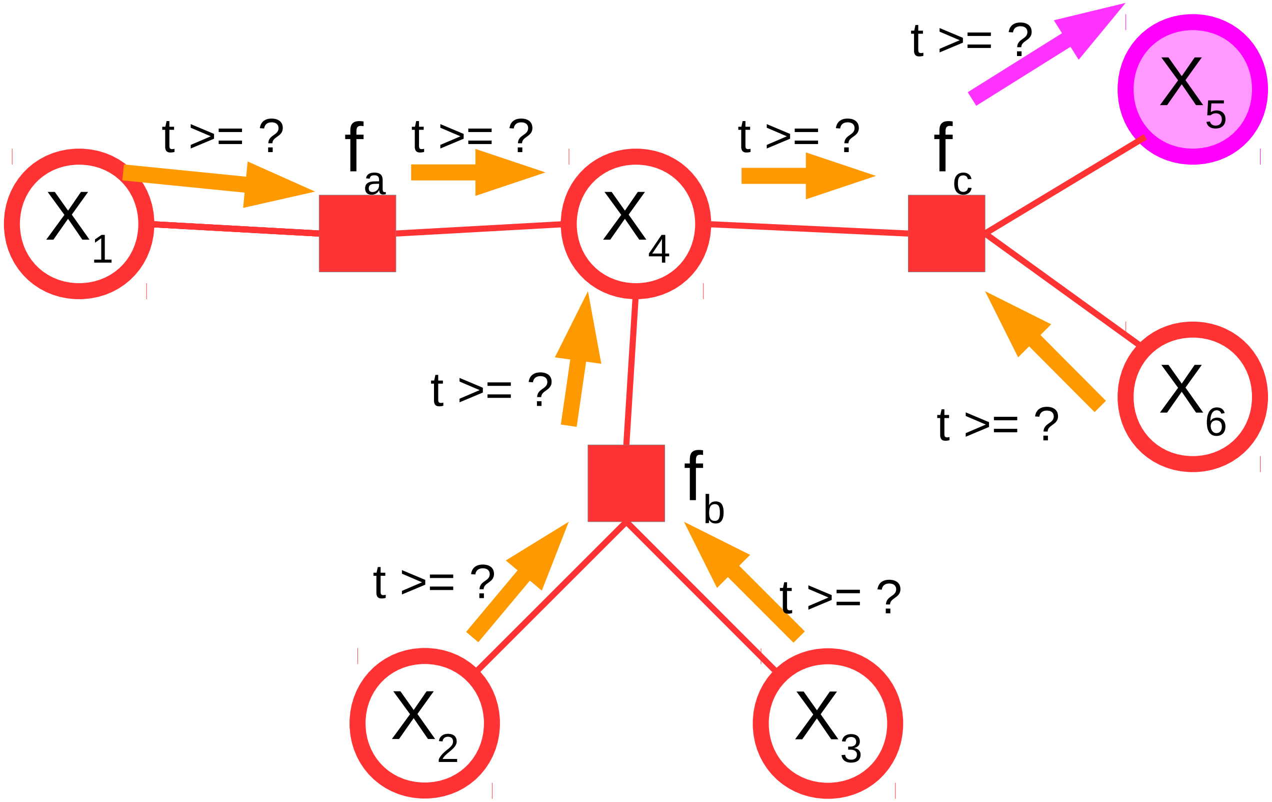
* This only holds for trees!

Second insight



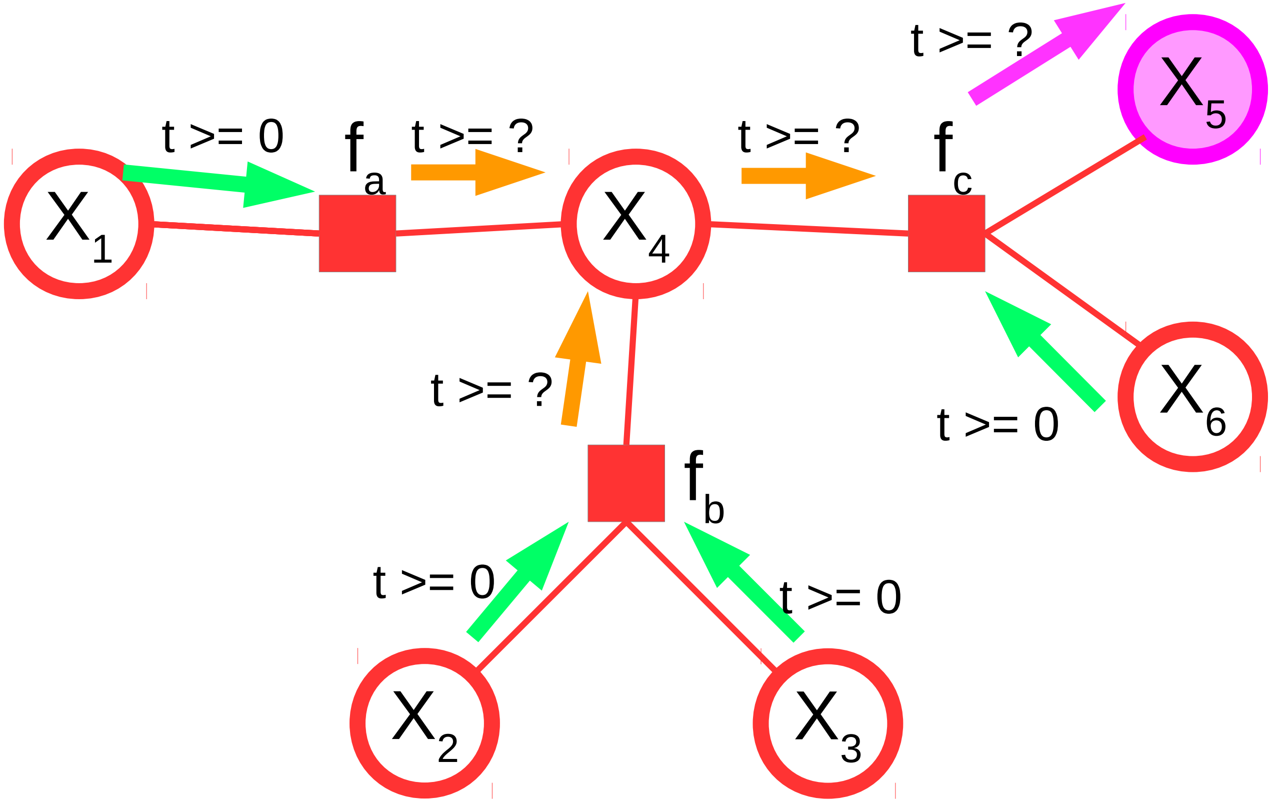
Let h be the height of the tree rooted at one message.

- At iteration h of BP, the message reaches its final value.



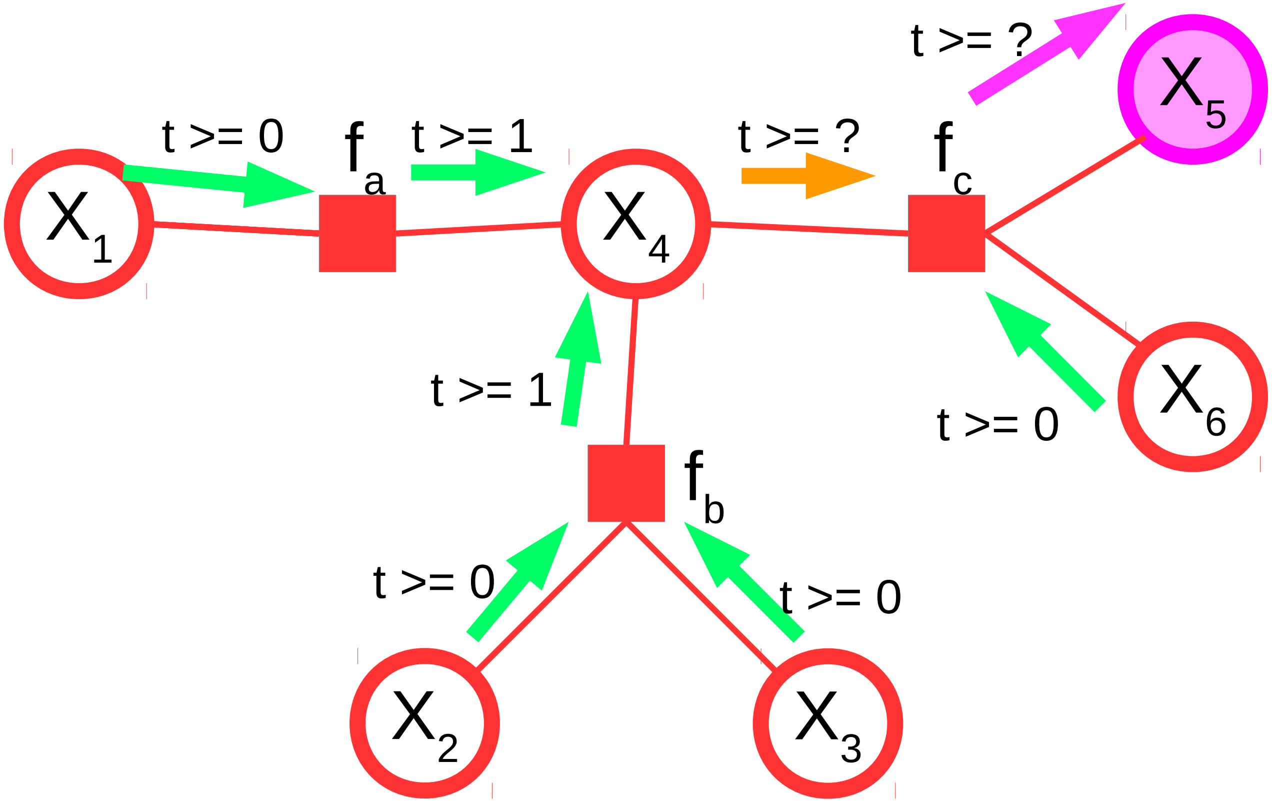
$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f' \in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X' \in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



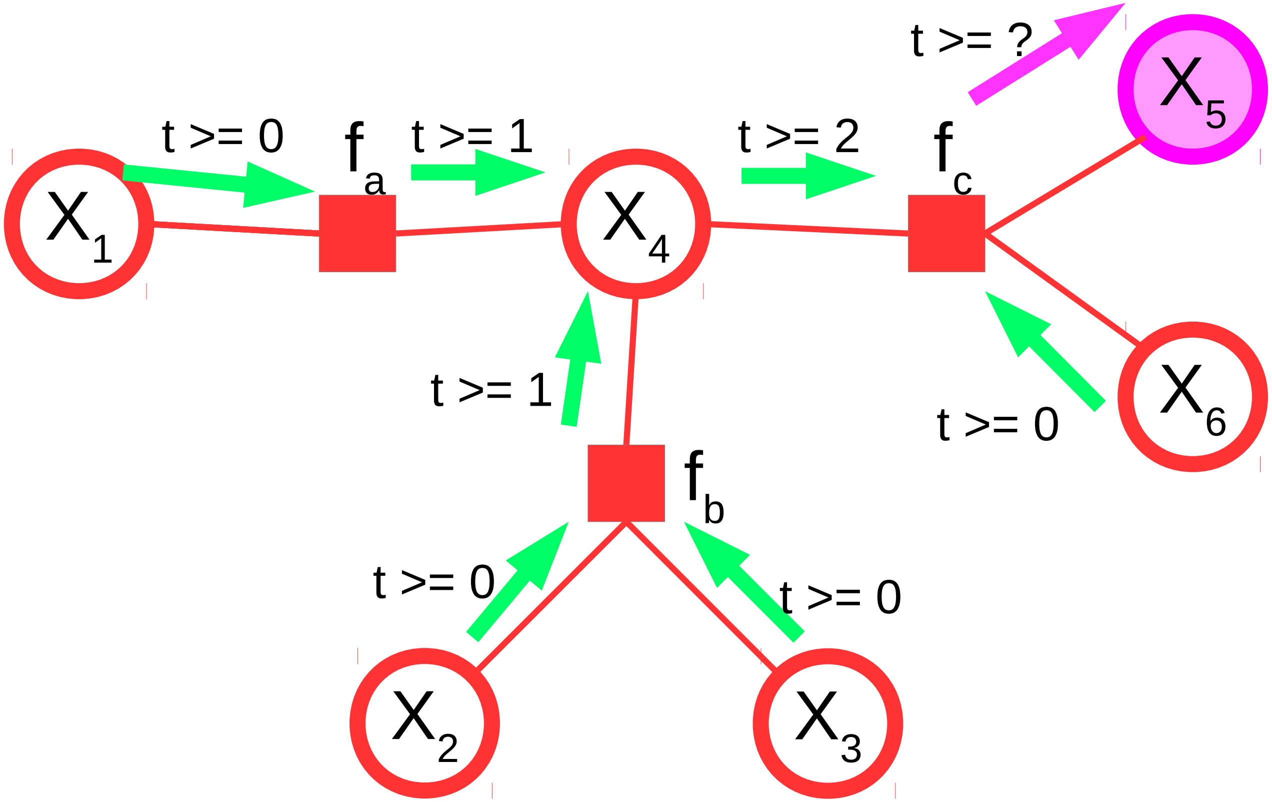
$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



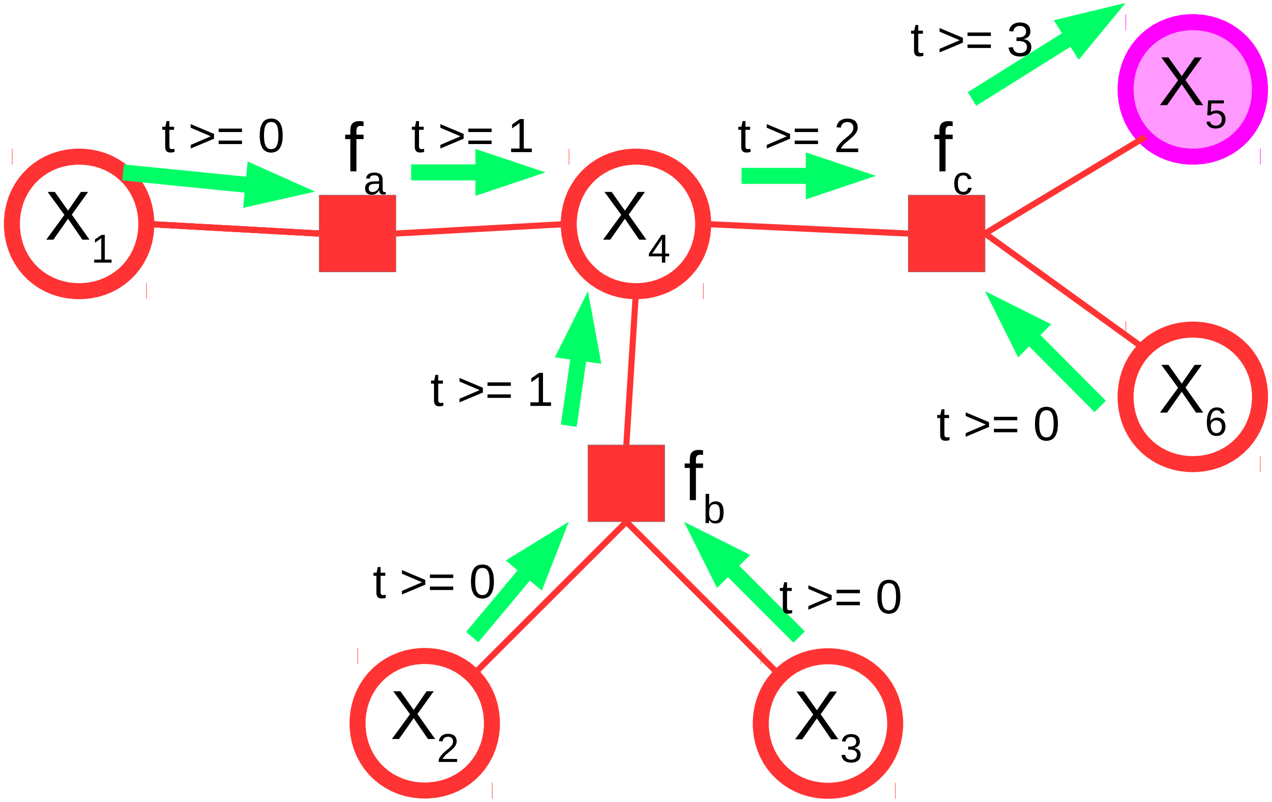
$$\mu_{X\to f}^{(t)}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu_{f'\to X}^{(t-1)}(\hat{x})$$

$$\mu_{f\to X}^{(t)}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu_{X'\to f}^{(t-1)}(\hat{x}')$$



$$\mu^{(t)}_{X\to f}(\hat{x}) = \prod_{f'\in N(X)\setminus\{f\}} \mu^{(t-1)}_{f'\to X}(\hat{x})$$

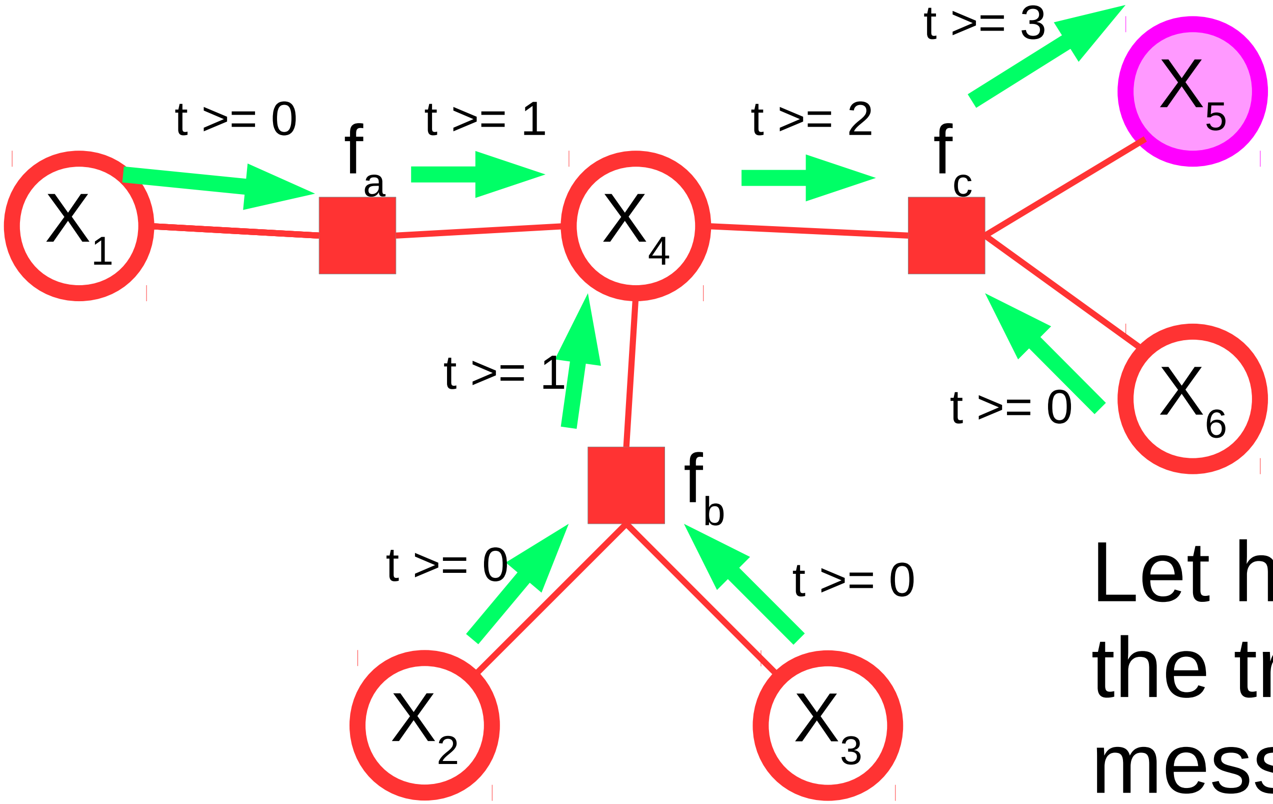
$$\mu^{(t)}_{f\to X}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X'\in N(f)\setminus\{X\}} \mu^{(t-1)}_{X'\to f}(\hat{x}')$$



$$\mu_{X\to f}^{(t)}(\hat{x}) = \prod_{f' \in N(X)\setminus\{f\}} \mu_{f'\to X}^{(t-1)}(\hat{x})$$

$$\mu_{f\to X}^{(t)}(\hat{x}) = \sum_{\mathbf{x}} f(\hat{\mathbf{x}}, \hat{x}) \prod_{X' \in N(f)\setminus\{X\}} \mu_{X'\to f}^{(t-1)}(\hat{x}')$$

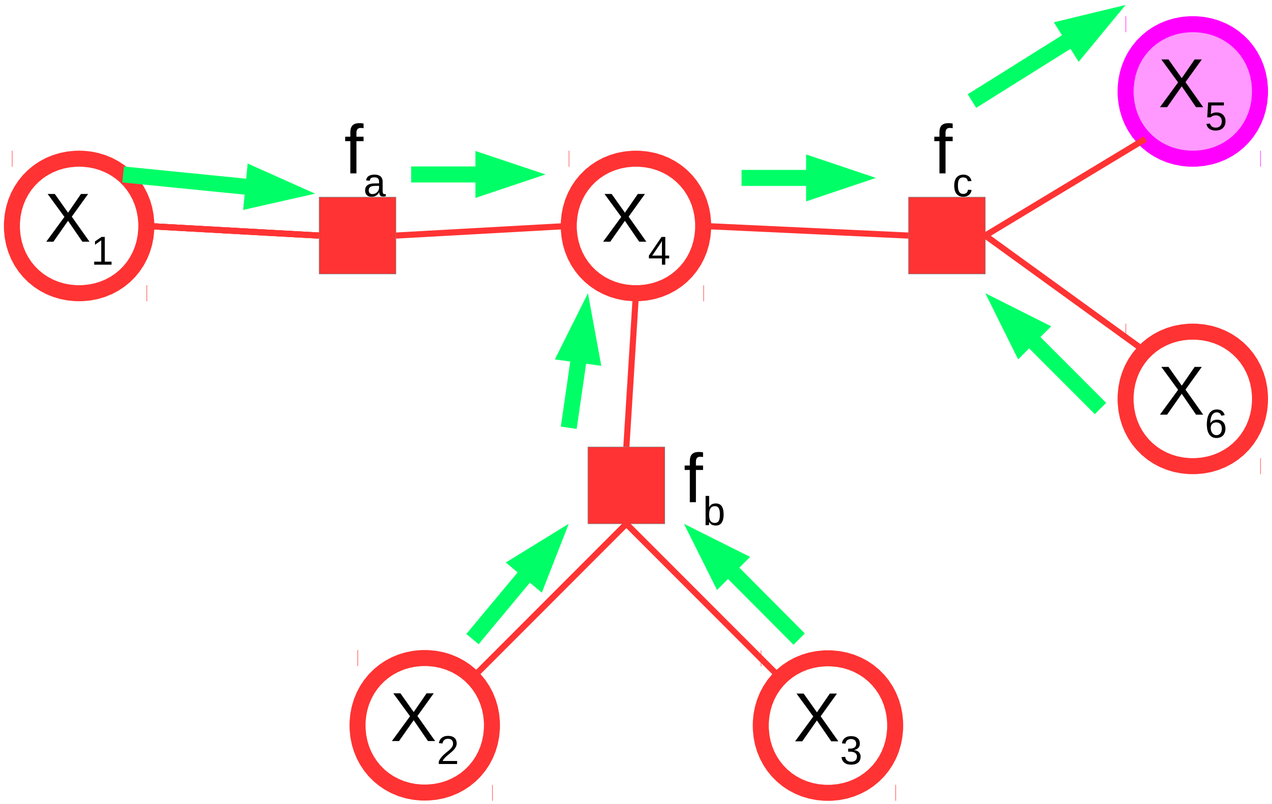
Second insight



Let h be the height of the tree rooted at one message.

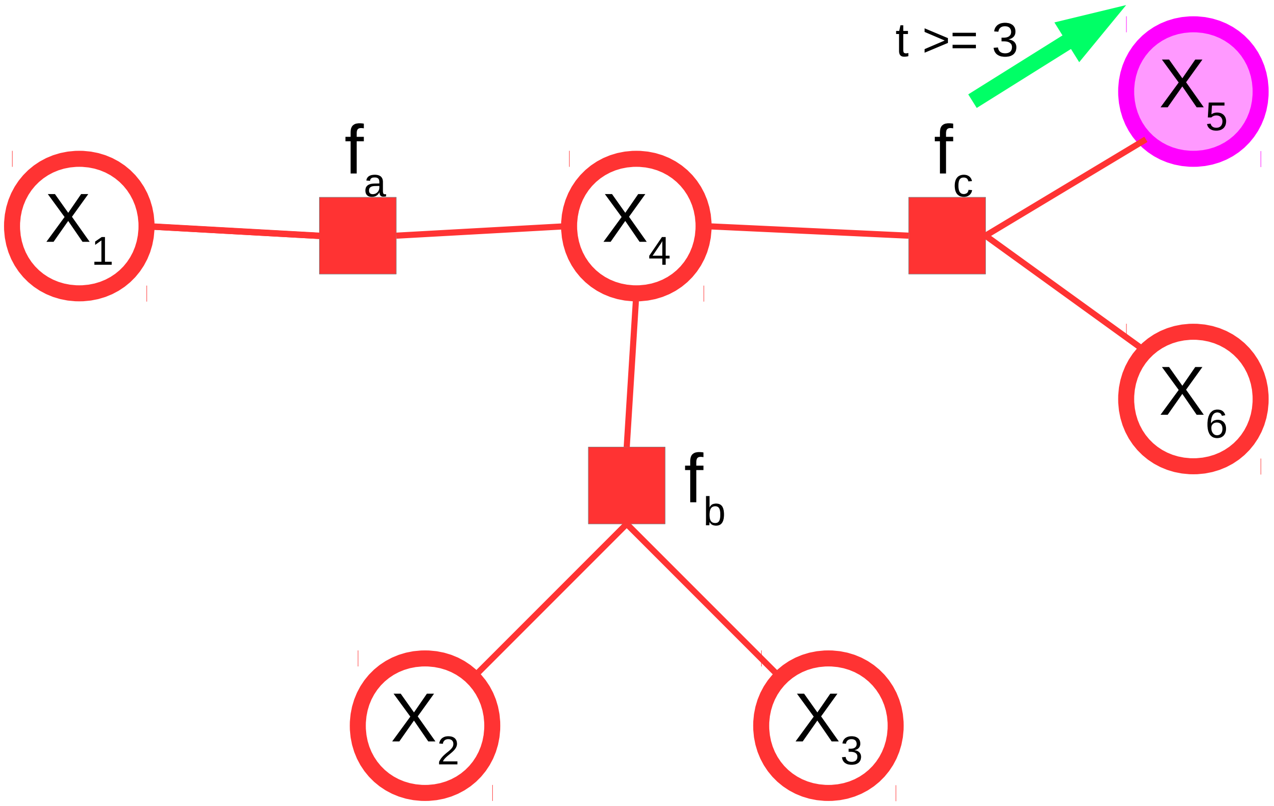
- At iteration h, the message reaches its final value.

Third insight

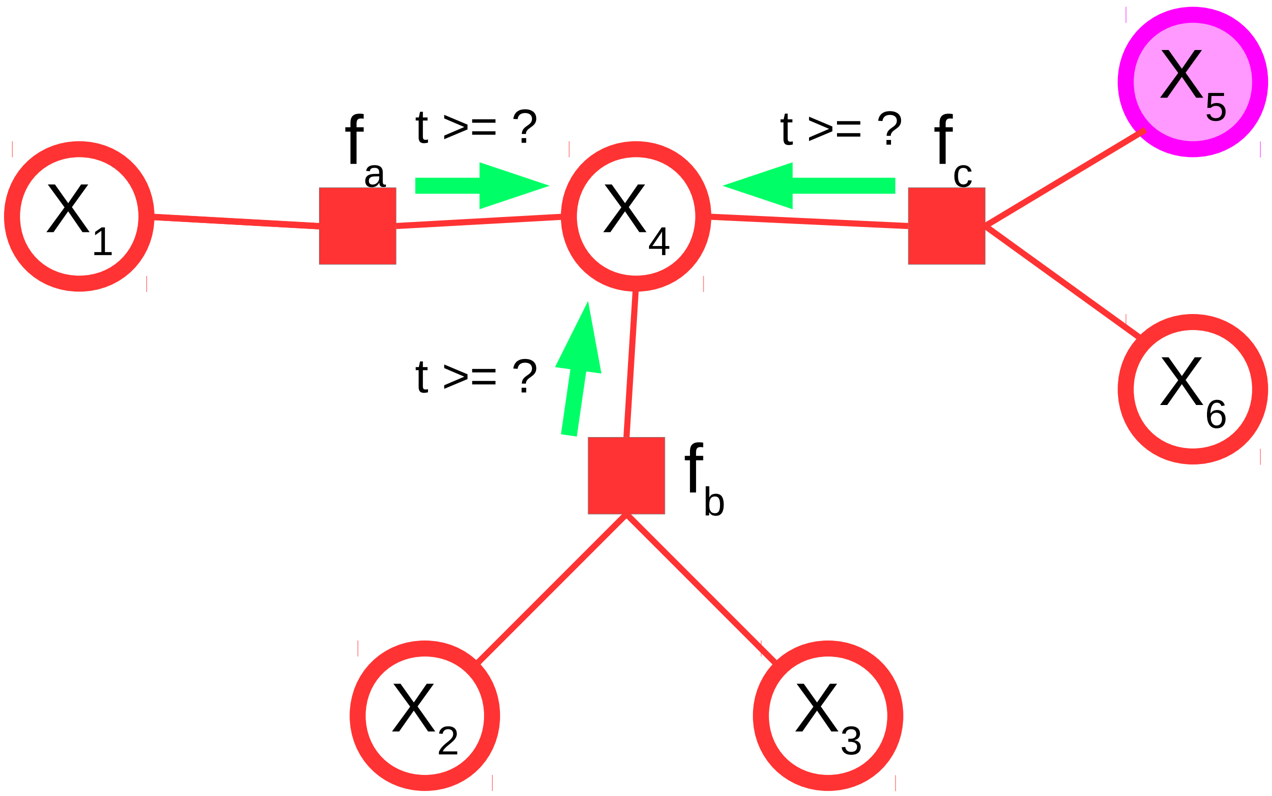


Let t_0 be the time at which all messages have reached their true value.
For any $t \geq t_0$, $P(X = \hat{x}) = \frac{1}{Z} \prod_{f \in N(X)} \mu^{(t)}_{f \to X}(\hat{x})$.

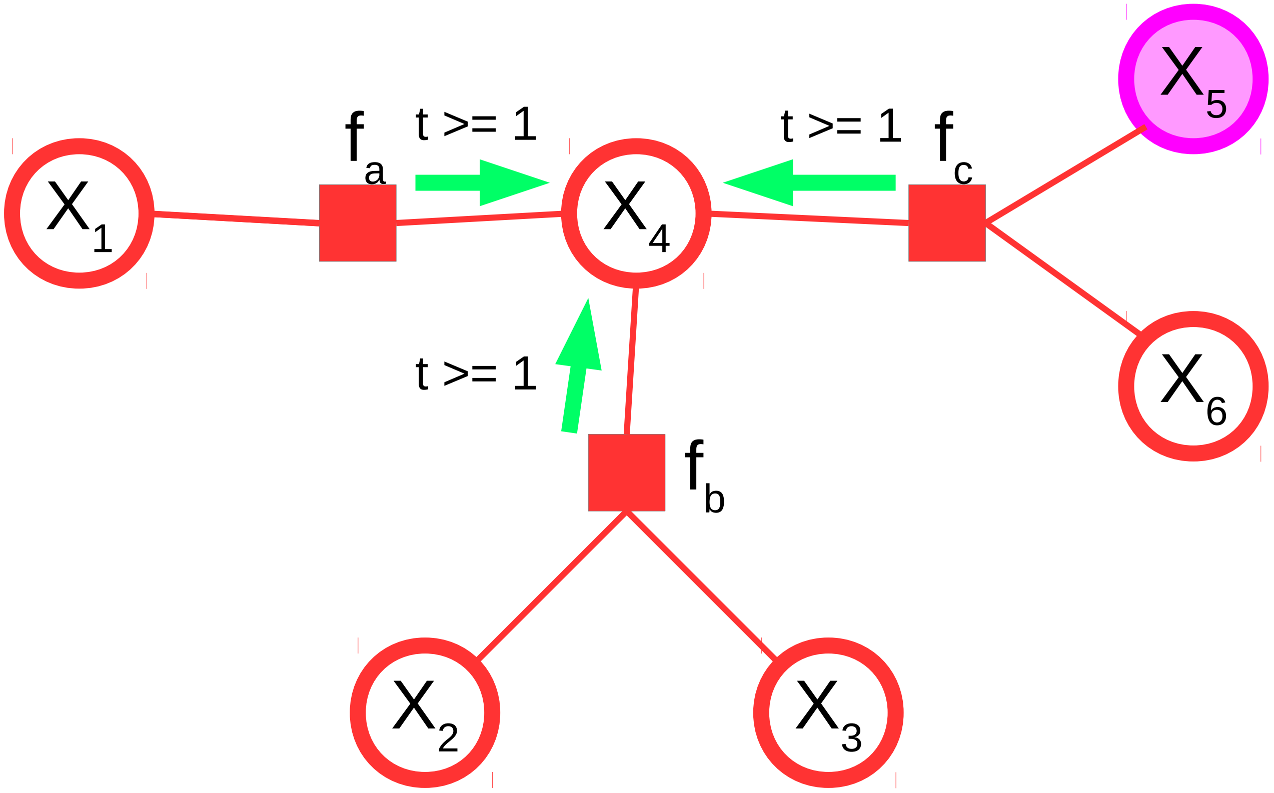
$P(X_5 = x_5)$



$P(X_4 = x_4)$



$P(X_4 = x_4)$



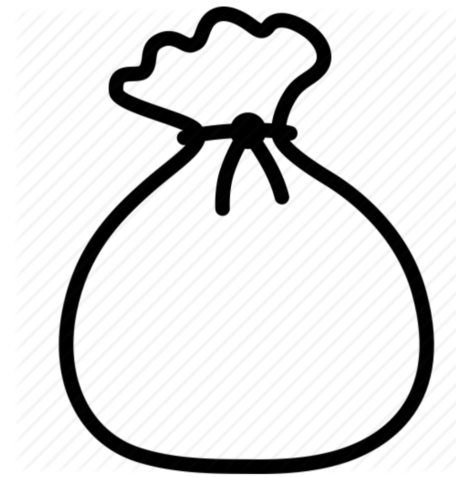
ETH zürich

Rejection sampling and MCMC

Carlos Cotrini
November 3, 2017

Probabilistic foundations of artificial intelligence

Computing expected loss from a burglary



What is the expected total value of items stolen?

Computing expected loss from a burglary

Given…

- Col, a collection of objects.
- Cap, the capacity of the bag.

- Let $\mathsf{FIT} := \{A \subseteq \mathsf{Col} \mid \sum_{b \in A} \mathit{weight}(b) \leq \mathsf{Cap}\}$

- Let B ~ Unif(FIT). That is,

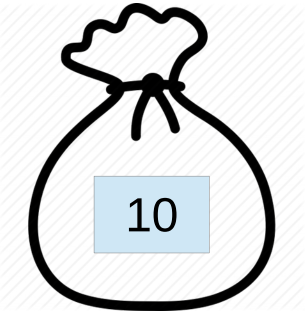
P(B = A) = 1/|FIT|

Estimate

$$\mathbb{E}\left[\sum_{b \in B} \mathit{value}(b)\right].$$

How about rejection sampling?


©2016 Sony Interactive Entertainment Inc. All rights reserved.
Design and specifications are subject to change without notice.

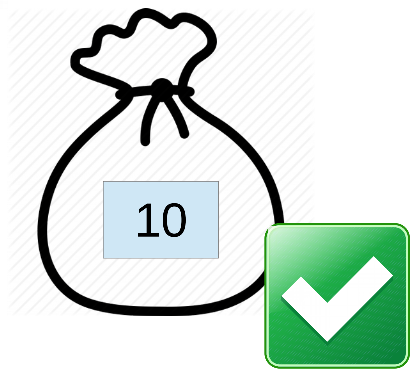

10


10

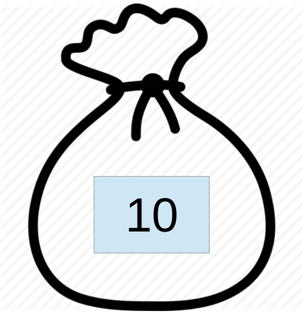

10


10

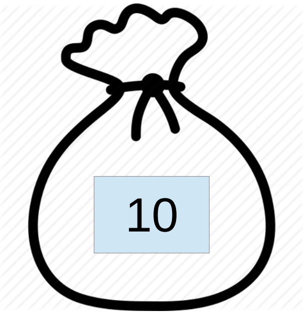

©2016 Sony Interactive Entertainment Inc. All rights reserved.
Design and specifications are subject to change without notice.


10

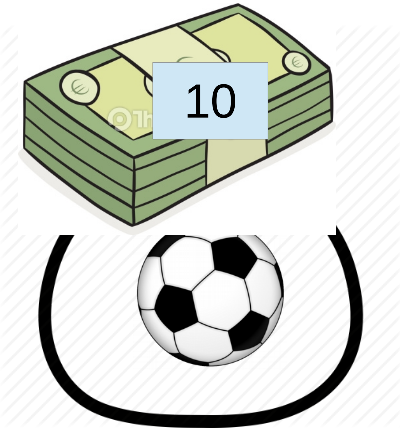

©2016 Sony Interactive Entertainment Inc. All rights reserved.
Design and specifications are subject to change without notice.


10


©2016 Sony Interactive Entertainment Inc. All rights reserved.
Design and specifications are subject to change without notice.

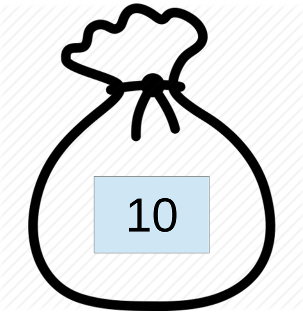

10


840

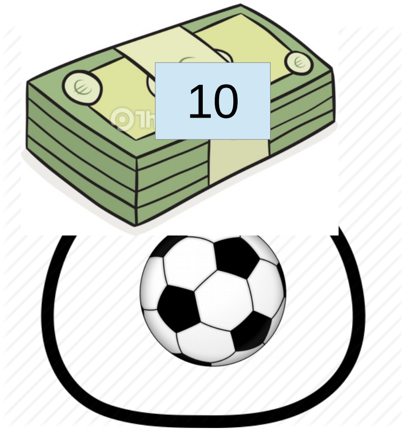

10




©2016 Sony Interactive Entertainment Inc. All rights reserved.
Design and specifications are subject to change without notice.



10

1/2(

840
+

10
)

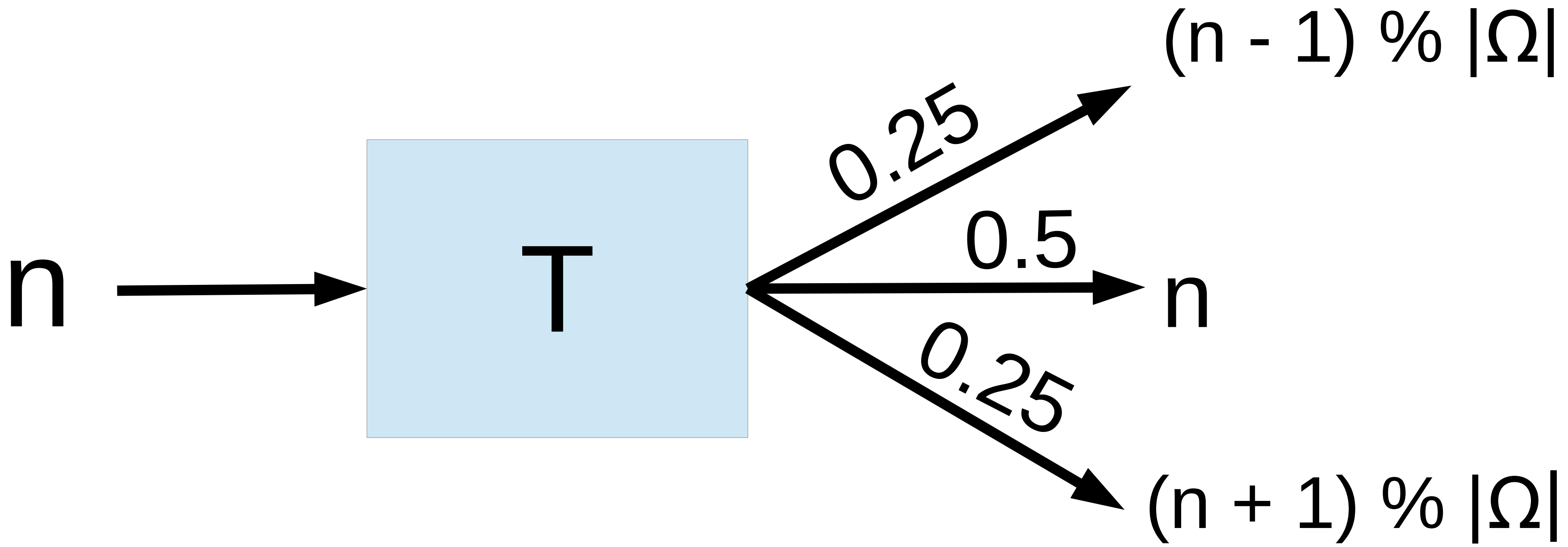
Let’s implement it

Generating samples from Q using MCMC

- Ingredients:
  - A prob. algo. T that transforms one sample into another.
  - A proof that T is “good” for Q.
- Recipe:
  - Take any sample x (not necessarily random).
  - For N sufficiently large, let x’ = $T^N(x) = T(...(T(x))…)$.
  - Return x’

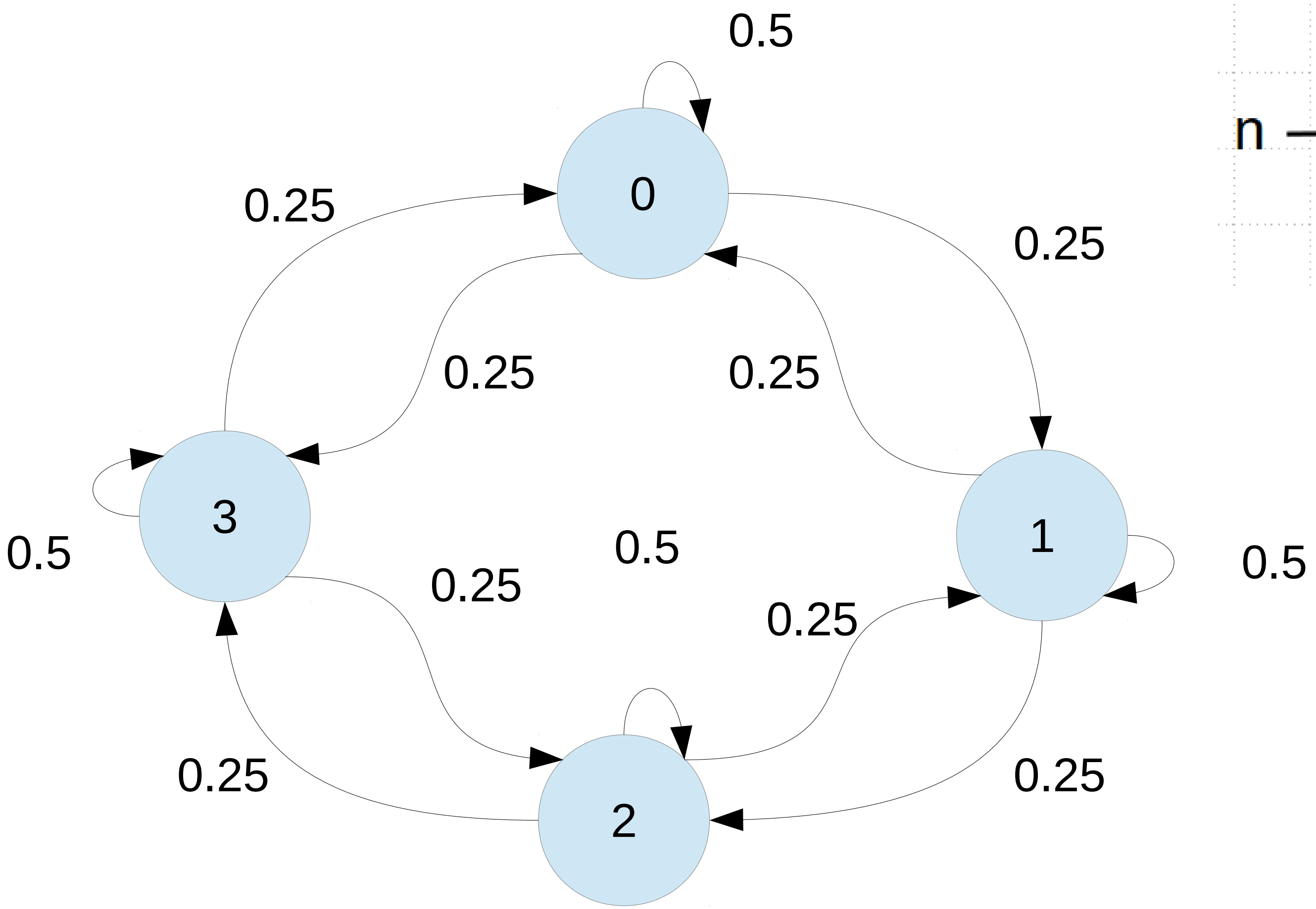
What makes T “good” for Q?

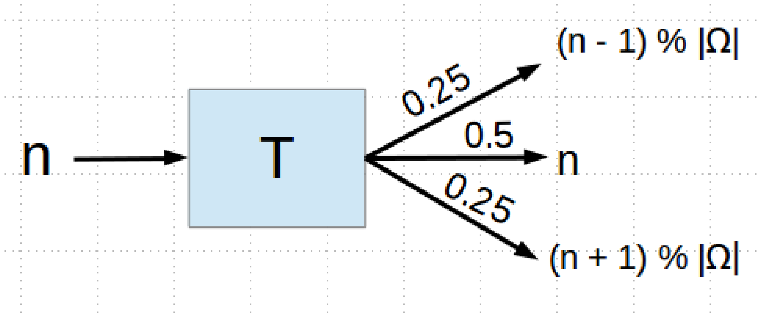
- Let Ω = {0,1,2,3} and Q = Unif(Ω).



What makes T "good" for Q?

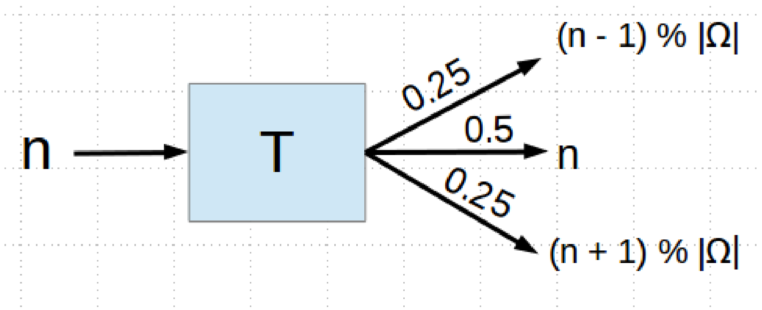
- Let Ω = {0,1,2,3} and Q = Unif(Ω).

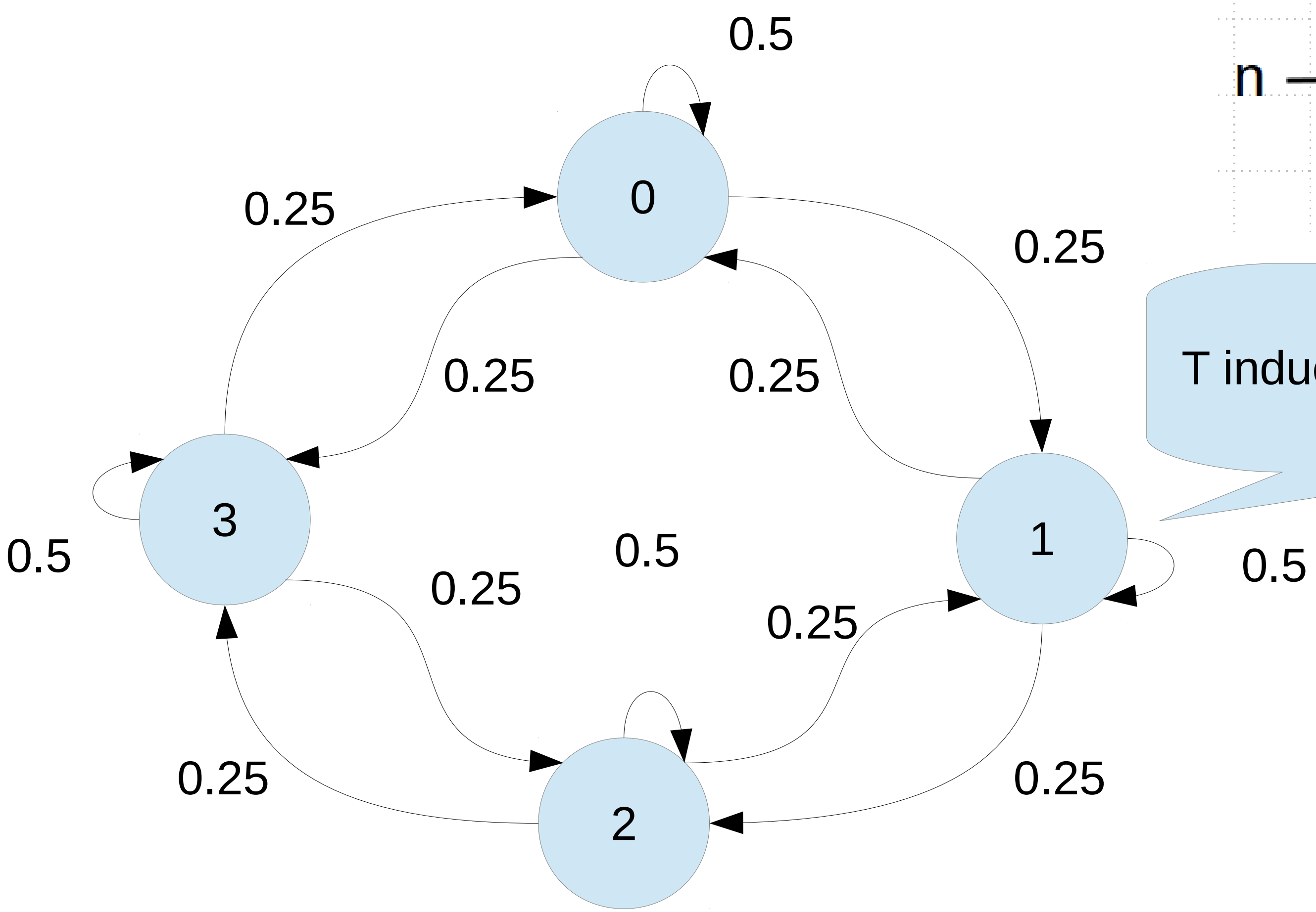




What makes T “good” for Q?

- Let Ω = {0,1,2,3} and Q = Unif(Ω).

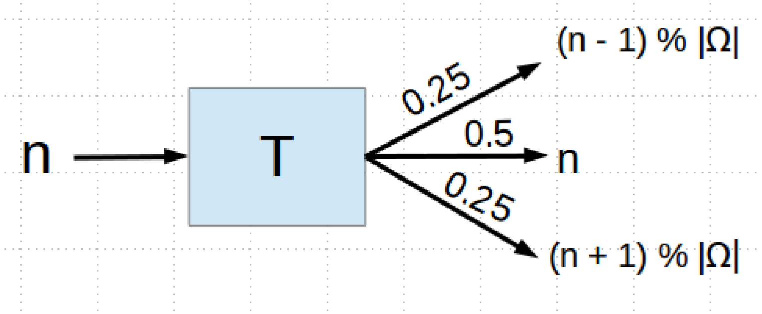




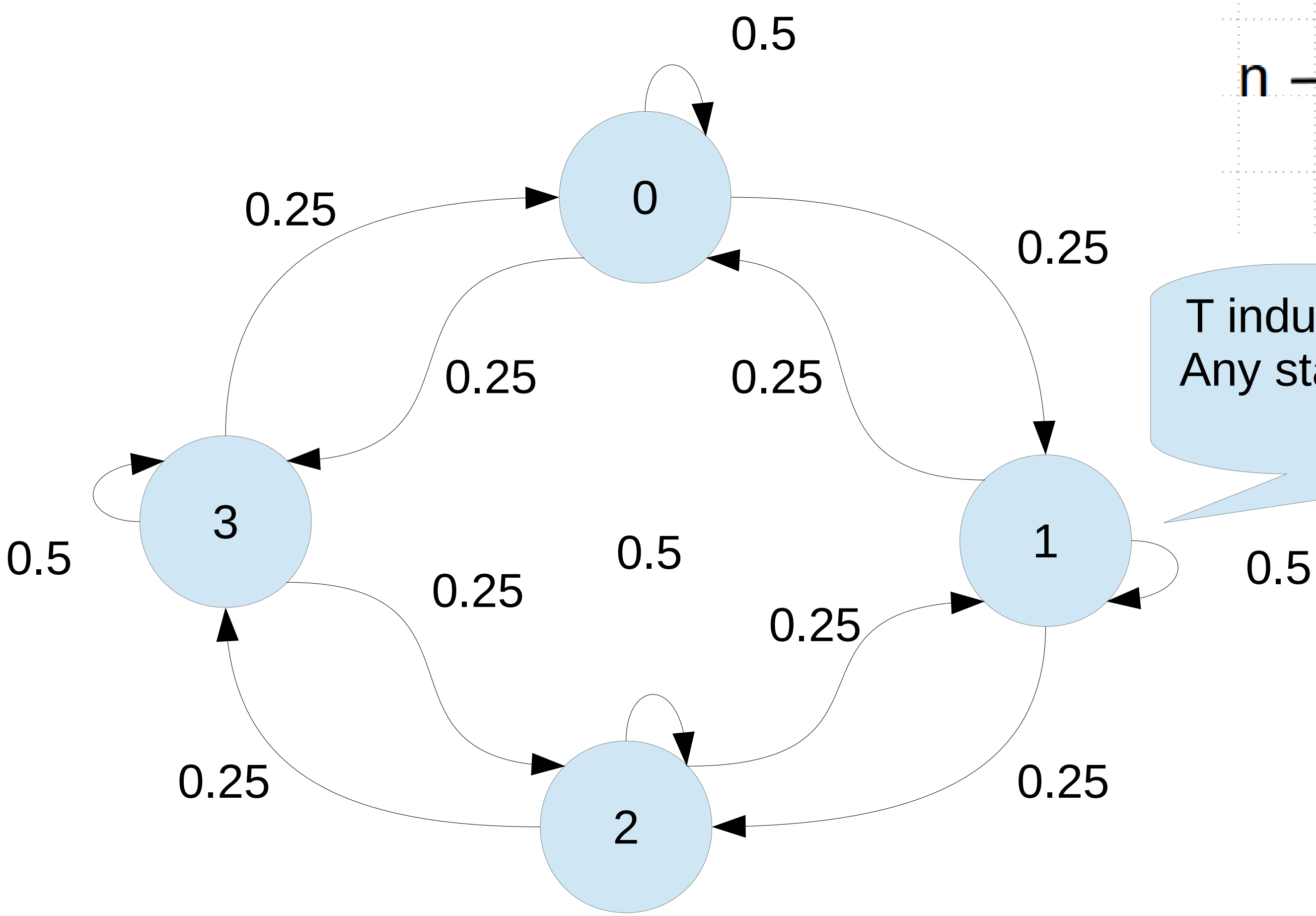
T induces an ergodic Markov chain!

What makes T "good" for Q?

- Let Ω = {0,1,2,3} and Q = Unif(Ω).

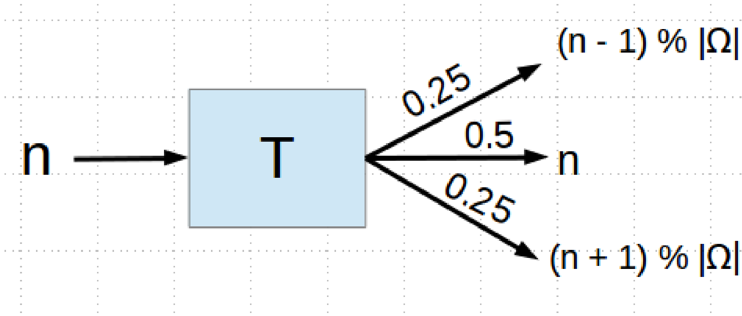


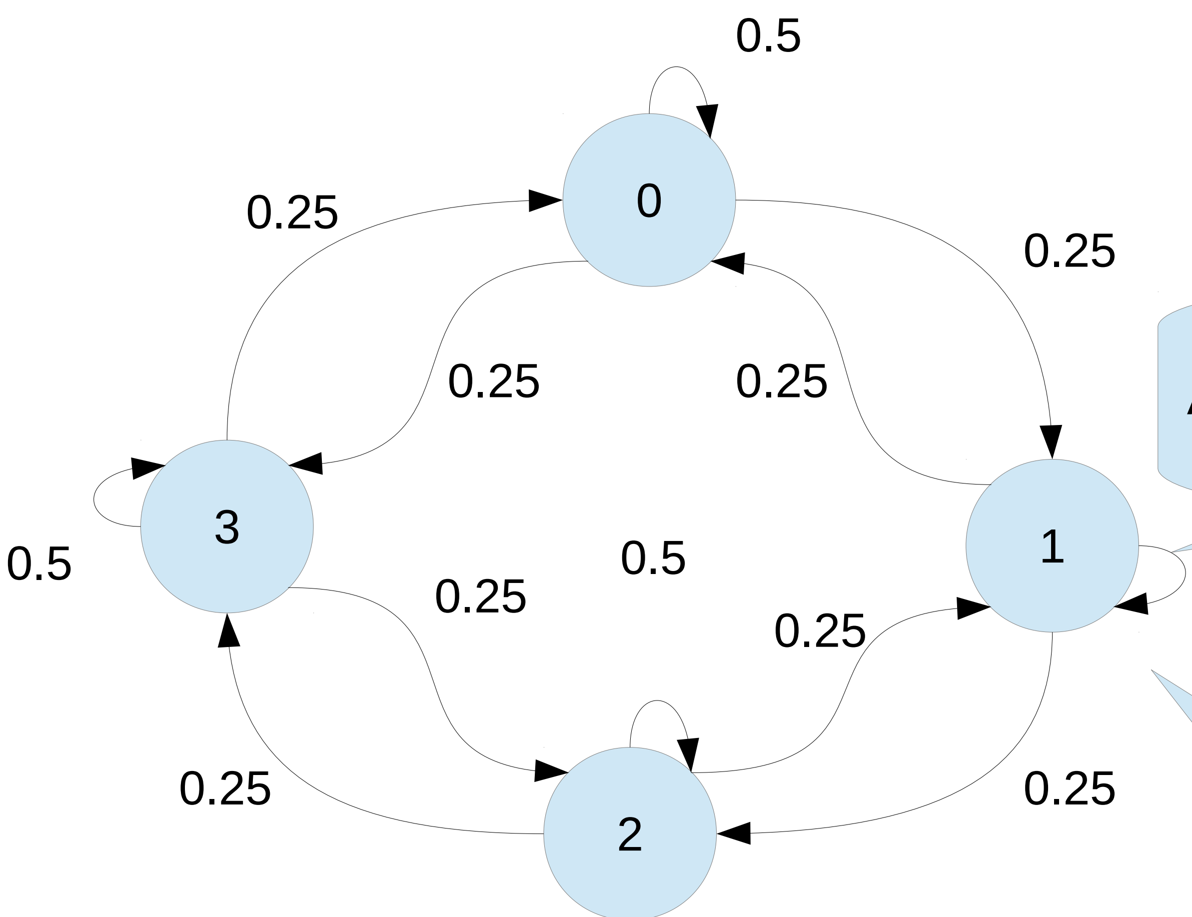
T induces an ergodic Markov chain!
Any state can reach any other state
In 2,000,000 steps.



What makes T “good” for Q?

- Let Ω = {0,1,2,3} and Q = Unif(Ω).





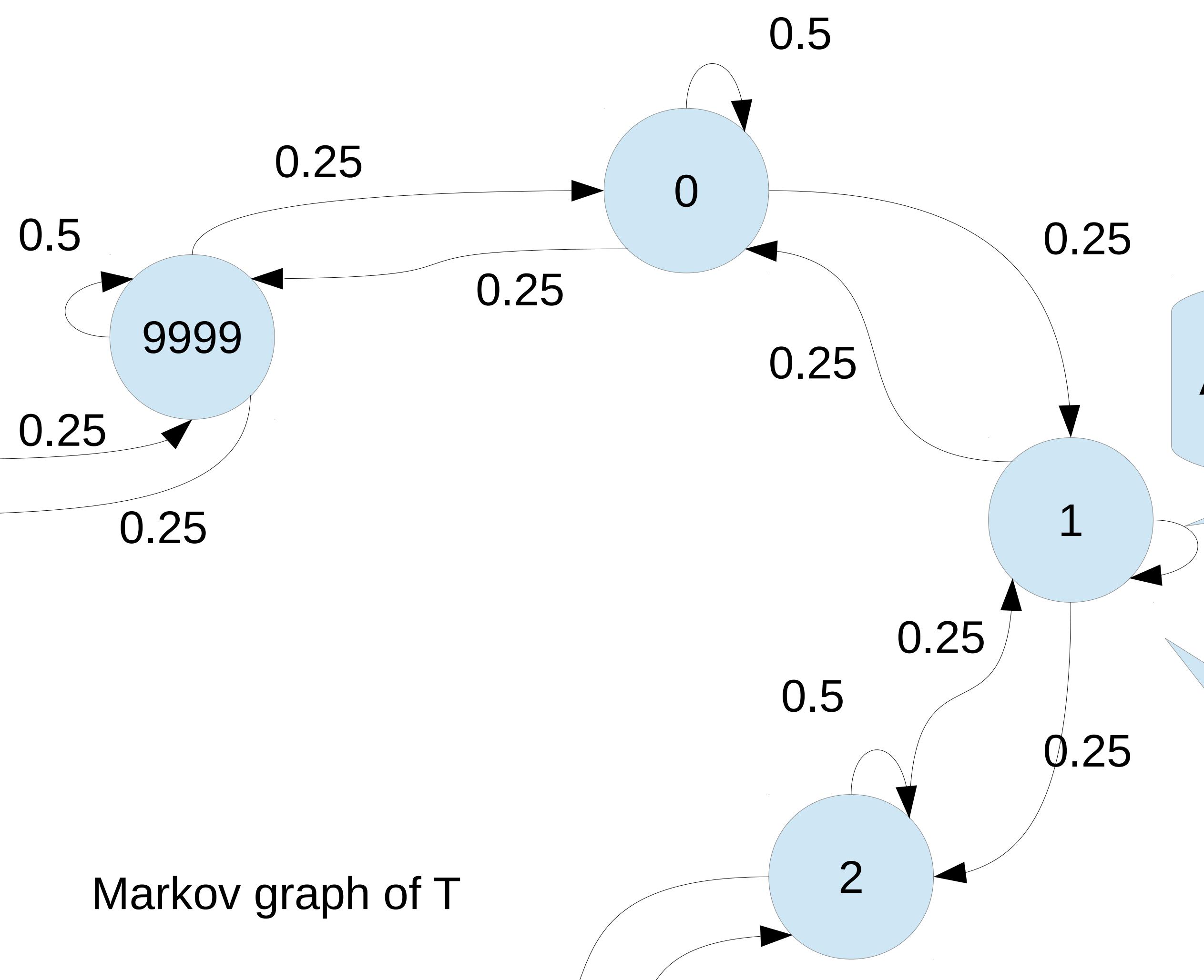
Markov graph of T

T induces an ergodic Markov chain!
Any state can reach any other state
In 2,000,000 steps.

Q(x) R(x | x’) = Q(x’) R(x’ | x),
for any two states x and x’.

What makes T “good” for Q?

- Let **Ω = {0,…, 9999}** and Q = Unif(Ω).



T induces an ergodic Markov chain!
Any state can reach any other state
In 2,000,000 steps.

Q(x) R(x | x’) = Q(x’) R(x’ | x),
for any two states x and x’.

Markov graph of T

What makes T "good" for Q?

- Let M be the Markov chain induced by T and let R be M's transition probability.
- T is "good" for Q if
  - M is ergodic.
  - $Q(x)R(x'|x) = Q(x')R(x'|x)$, for all x, x'.

What makes T “good” for Q?

- Let M be the Markov chain induced by T and let R be M’s transition probability.
- T is “good” for Q if
  - M is ergodic.
  - Q(x)R(x’|x) = Q(x’)R(x’|x), for all x, x’.

Warning, these are sufficiency conditions! There may be other algorithms that are “good” for Q, but do not satisfy these conditions.

Computing expected loss from a burglary

Generating samples from a complex distribution Q using MCMC

- Ingredients:
  - A prob. algo. T that transforms one sample into another.
  - A proof that T is “good” for Q.
- Recipe:
  - Take any sample x (not necessarily random).
  - For N sufficiently large, let x’ = $T^N(x) = T(...(T(x))…)$.
  - Return x’

A “good” T for the uniform distr. on 2^{Col}

- Let B in FIT.
  - Flip a coin. If heads, then return B.
  - Pick an object b in Col uniformly at random.
  - If b in B:
    - return B \ {b}
  - If b not in B:
    - If the total weight of B U {b} <= Cap:
      - return B U {b}
    - Else:
      - return B

Proving T is "good" for Q

- Part 1 of 2. Show T induces an ergodic Markov chain.

Proving T is “good” for Q

- Part 1 of 2. Show T induces an ergodic Markov chain.
  - Insight 1: The Markov graph of T is connected.
    - If you are lucky enough, T transforms any B into the empty set after some steps. If you are even luckier, then T transforms the empty set into B’ after some steps.

Proving T is “good” for Q

- Part 1 of 2. Show T induces an ergodic Markov chain.
  - Insight 1: The Markov graph of T is connected.
    - If you are lucky enough, T transforms any B into the empty set after some steps. If you are even luckier, then T transforms the empty set into B’ after some steps.
  - Insight 2: If L >> $|2^{Col}|$, then T can reach any B’ from any B in at most L steps.

Proving T is "good" for Q

- Part 1 of 2. Show T induces an ergodic Markov chain.
  - Insight 1: The Markov graph of T is connected.
    - If you are lucky enough, T transforms any B into the empty set after some steps. If you are even luckier, then T transforms the empty set into B' after some steps.
  - Insight 2: If L >> $|2^{Col}|$, then T can reach any B' from any B in at most L steps.
  - Insight 3: There is a self-loop for every B' in the Markov graph.
    - If you happen to arrive to B' before t steps, just use the extra steps on the self-loop to arrive in exactly t steps.

Proving T is “good” for Q

- Part 2 of 2. $Q(x)\, R(x \mid x') = Q(x')\, R(x' \mid x)$, for any x, x’.

ETH zürich

Proving T is “good” for Q

- Part 2 of 2. $Q(x) R(x \mid x') = Q(x') R(x' \mid x)$, for any x, x’.
  - Hint 1: $Q(x) = Q(x')$.

Proving T is "good" for Q

- Part 2 of 2. $Q(x)\, R(x \mid x') = Q(x')\, R(x' \mid x)$, for any x, x'.
  - Hint 1: $Q(x) = Q(x')$.
  - Hint 2: $R(x' \mid x) = R(x' \mid x)$.