

# Soft Constraints: Exponential Models

Factor graphs (undirected graphical models) and their connection to constraint programming

600.325/425 Declarative Methods - J. Eisner 1

# Soft constraint problems (e.g. MAX-SAT)

- Given
  - n variables
  - m constraints, over various subsets of variables
- Find
  - Assignment to the n variables that maximizes the number of satisfied constraints.

600.325/425 Declarative Methods - J. Eisner 2

# Soft constraint problems (e.g. MAX-SAT)

- Given
  - n variables
  - m constraints, over various subsets of variables
  - m weights, one per constraint**
- Find
  - Assignment to the n variables that maximizes the **total weight** of the satisfied constraints.
    - Equivalently, minimizes total weight of violated constraints.

600.325/425 Declarative Methods - J. Eisner 3

# Draw problem structure as a “factor graph”

Each constraint (“factor”) is a function of the values of its variables.

weight  $w \rightarrow$   
if satisfied, factor= $\exp(w)$   
if violated, factor=1

- Measure goodness of an assignment by the **product of all the factors** ( $\geq 0$ ).
  - How can we reduce previous slide to this?
    - There, each constraint was either satisfied or not (simple case).
    - There, good score meant large total weight for satisfied constraints.

figure thanks to Brian Potetz 600.325/425 Declarative Methods - J. Eisner 4

# Draw problem structure as a “factor graph”

Each constraint (“factor”) is a function of the values of its variables.

weight  $w \rightarrow$   
if satisfied, factor=1  
if violated, factor= $\exp(-w)$

- Measure goodness of an assignment by the **product of all the factors** ( $\geq 0$ ).
  - How can we reduce previous slide to this?
    - There, each constraint was either satisfied or not (simple case).
    - There, good score meant **small** total weight for **violated** constraints.

figure thanks to Brian Potetz 600.325/425 Declarative Methods - J. Eisner 5

# Draw problem structure as a “factor graph”

Each constraint (“factor”) is a function of the values of its variables.

- Measure goodness of an assignment by the **product of all the factors** ( $\geq 0$ ).
- Models like this show up **all the time**.

figure thanks to Brian Potetz 600.325/425 Declarative Methods - J. Eisner 6

### Example: Ising Model

(soft version of graph coloring, on a grid graph)

Model	Physics
Boolean vars	Magnetic polarity at points on the plane
Binary equality constraints	?
Unary constraints	?
MAX-SAT	?

Figure thanks to ???

600.325/425 Declarative Methods - J. Eisner 7

### Example: Parts of speech

(or other sequence labeling problems)

Or, if the input words are given, you can customize the factors to them:

600.325/425 Declarative Methods - J. Eisner 8

### Local factors in a graphical model

- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging

Possible tagging (i.e., assignment to remaining variables)

Observed input sentence (shaded)

9

### Local factors in a graphical model

- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging

Possible tagging (i.e., assignment to remaining variables)

Another possible tagging

Observed input sentence (shaded)

10

### Local factors in a graphical model

- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging

"Binary" factor that measures compatibility of 2 adjacent tags

Model reuses same parameters at this position

11

### Local factors in a graphical model

- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging

"Unary" factor evaluates this tag

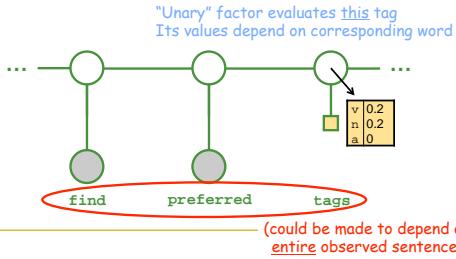
Its values depend on corresponding word

tags can't be adj

12

## Local factors in a graphical model

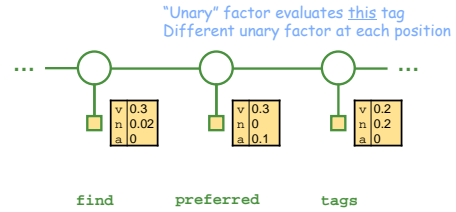
- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging



13

## Local factors in a graphical model

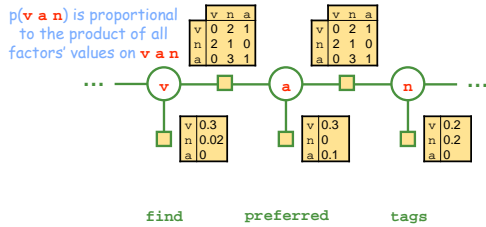
- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging



14

## Local factors in a graphical model

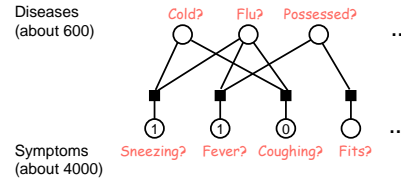
- First, a familiar example
  - Conditional Random Field (CRF) for POS tagging



15

## Example: Medical diagnosis (QMR-DT)

- Patient is sneezing with a fever; no coughing

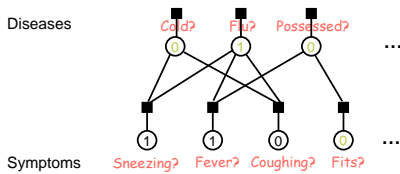


600.325/425 Declarative Methods - J. Eisner

16

## Example: Medical diagnosis

- Patient is sneezing with a fever; no coughing
  - Possible diagnosis: Flu (without coughing)
    - But maybe it's not flu season ...

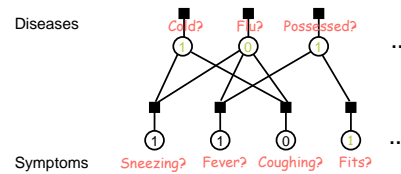


600.325/425 Declarative Methods - J. Eisner

17

## Example: Medical diagnosis

- Patient is sneezing with a fever; no coughing
  - Possible diagnosis: Cold (without coughing), and possessed (better ask about fits ...)



600.325/425 Declarative Methods - J. Eisner

18

### Example: Medical diagnosis

- Patient is sneezing with a fever; no coughing
  - Possible diagnosis: Spontaneous sneezing, and possessed (better ask about fits ...)

Note: Here symptoms & diseases are boolean. We could use real #s to denote degree.

600.325/425 Declarative Methods - J. Eisner 19

### Example: Medical diagnosis

- What are the factors, exactly?
- Factors that are w or 1 (weighted MAX-SAT):

- If observe sneezing, get a disjunctive clause (Human v Cold v Flu)
- If observe non-sneezing, get unit clauses ( $\neg$ Human)  $\wedge$  ( $\neg$ Cold)  $\wedge$  ( $\neg$ Flu)

600.325/425 Declarative Methods - J. Eisner 20

### Example: Medical diagnosis

- What are the factors, exactly?
- Factors that are probabilities:

Use a little "noisy OR" model here:  
 $x = (\text{Human}, \text{Cold}, \text{Flu})$ , e.g., (1,1,0). More 1's should increase  $p(\text{sneezing})$ .  
 $p(\neg\text{sneezing} | x) = \exp(-w \cdot x)$  e.g.,  $w = (0.05, 2, 5)$

Would get logistic regression model if we replaced  $\exp$  by sigmoid, i.e.,  $\exp/(1+\exp)$

600.325/425 Declarative Methods - J. Eisner 21

### Example: Medical diagnosis

- What are the factors, exactly?
- Factors that are probabilities:

- If observe sneezing, get a factor  $(1 - \exp(-w \cdot x))$   $(1 - 0.95^{\text{Human}} 0.14^{\text{Cold}} 0.007^{\text{Flu}})$
- If observe non-sneezing, get a factor  $\exp(-w \cdot x)$   $0.95^{\text{Human}} 0.14^{\text{Cold}} 0.007^{\text{Flu}}$

As  $w \rightarrow \infty$ , approach Boolean case (product of all factors  $\rightarrow 1$  if SAT, 0 if UNSAT)

600.325/425 Declarative Methods - J. Eisner 22

### Technique #1: Branch and bound

- Exact backtracking technique we've already studied.
  - And used via ECLIPSe's "minimize" routine.
- Propagation can help prune branches of the search tree (add a hard constraint that we must do better than best solution so far).
- Worst-case exponential.

600.325/425 Declarative Methods - J. Eisner 23

### Technique #2: Variable Elimination

- Exact technique we've studied; worst-case exponential.

Bucket E:  $E \neq D, E \neq C$   
 Bucket D:  $D \neq A, D = C$   
 Bucket C:  $C \neq B, A \neq C$   
 Bucket B:  $B \neq A, B \neq A$   
 Bucket A: contradiction

join all constraints in E's bucket yielding a new constraint on D (and C)  
 now join all constraints in D's bucket ...

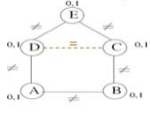
- But how do we do it for soft constraints?
  - How do we join soft constraints?

figure thanks to Rina Dechter

600.325/425 Declarative Methods - J. Eisner 24

## Technique #2: Variable Elimination

- Easiest to explain via Dyna.

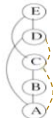
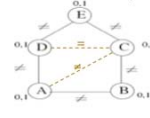


- goal max=  $f1(A,B)*f2(A,C)*f3(A,D)*f4(C,E)*f5(D,E)$ .  
tempE(C,D)

- tempE(C,D) max=  $f4(C,E)*f5(D,E)$ .  
to eliminate E, join constraints mentioning E, and project E out

## Technique #2: Variable Elimination

- Easiest to explain via Dyna.

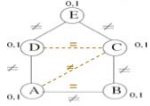


- goal max=  $f1(A,B)*f2(A,C)*f3(A,D)*tempE(C,D)$ .  
tempD(A,C)

- tempD(A,C) max=  $f3(A,D)*tempE(C,D)$ .  
tempE(C,D) max=  $f4(C,E)*f5(D,E)$ .  
to eliminate D, join constraints mentioning D, and project D out

## Technique #2: Variable Elimination

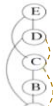
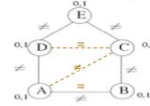
- Easiest to explain via Dyna.



- goal max=  $f1(A,B)*f2(A,C)*tempD(A,C)$ .  
tempC(A)
- tempC(A) max=  $f2(A,C)*tempD(A,C)$ .
- tempD(A,C) max=  $f3(A,D)*tempE(C,D)$ .
- tempE(C,D) max=  $f4(C,E)*f5(D,E)$ .

## Technique #2: Variable Elimination

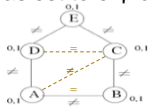
- Easiest to explain via Dyna.



- goal max= tempC(A)\*f1(A,B).  
tempB(A)
- tempB(A) max=  $f1(A,B)$ .
- tempC(A) max=  $f2(A,C)*tempD(A,C)$ .
- tempD(A,C) max=  $f3(A,D)*tempE(C,D)$ .
- tempE(C,D) max=  $f4(C,E)*f5(D,E)$ .

## Technique #2: Variable Elimination

- Easiest to explain via Dyna.



- goal max= tempC(A)\*tempB(A).
- tempB(A) max=  $f1(A,B)$ .
- tempC(A) max=  $f2(A,C)*tempD(A,C)$ .
- tempD(A,C) max=  $f3(A,D)*tempE(C,D)$ .
- tempE(C,D) max=  $f4(C,E)*f5(D,E)$ .

## Probabilistic interpretation of factor graph

("undirected graphical model")

Each factor is a function  $\geq 0$  of the values of its variables.

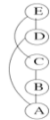
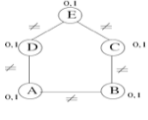


Measure goodness of an assignment by the product of all the factors.

- For any assignment  $x = (x_1, \dots, x_5)$ , define  $u(x)$  = product of all factors, e.g.,  $u(x) = f1(x)*f2(x)*f3(x)*f4(x)*f5(x)$ .
- We'd like to interpret  $u(x)$  as a probability distribution over all  $2^5$  assignments.
- Do we have  $u(x) \geq 0$ ? Yes. ☺
- Do we have  $\sum u(x) = 1$ ? No.  $\sum u(x) = Z$  for some  $Z$ . ☹
- So  $u(x)$  is not a probability distribution.
- But  $p(x) = u(x)/Z$  is!

## Z is hard to find ... (the "partition function")

- Exponential time with this Dyna program.

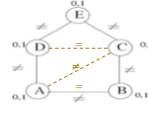


- goal  $\text{max} = f_1(A,B) * f_2(A,C) * f_3(A,D) * f_4(C,E) * f_5(D,E)$ .

This explicitly sums over all  $2^5$  assignments.  
We can do better by variable elimination ...  
(although still exponential time in worst case).  
Same algorithm as before: just replace  $\text{max} =$  with  $+=$ .

## Z is hard to find ... (the "partition function")

- Faster version of Dyna program, after var elim.



- goal  $+= \text{tempC}(A) * \text{tempB}(A)$ .
- $\text{tempB}(A) += f_1(A,B)$ .
- $\text{tempC}(A) += f_2(A,C) * \text{tempD}(A,C)$ .
- $\text{tempD}(A,C) += f_3(A,D) * \text{tempE}(C,D)$ .
- $\text{tempE}(C,D) += f_4(C,E) * f_5(D,E)$ .

## Why a probabilistic interpretation?

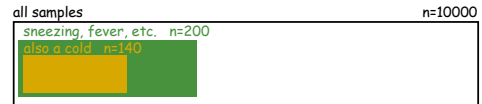
- Allows us to make **predictions**.
  - You're sneezing with a fever & no cough.
  - Then what is the *probability* that you have a cold?
- Important in **learning** the factor functions.
  - Maximize the probability of training data.
- Central to deriving fast **approximation** algorithms.
  - "Message passing" algorithms where nodes in the factor graph are repeatedly updated based on adjacent nodes.
  - Many such algorithms. E.g., survey propagation is the current best method for random 3-SAT problems. Hot area of research!

## Probabilistic interpretation → Predictions

You're sneezing with a fever & no cough.

Then what is the *probability* that you have a cold?

- Randomly sample 10000 assignments from  $p(x)$ .
- In 200 of them (2%), patient is sneezing with a fever and no cough.
- In 140 (1.4%) of those, the patient also has a cold.



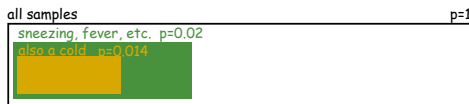
answer: 70% (140/200)

## Probabilistic interpretation → Predictions

You're sneezing with a fever & no cough.

Then what is the *probability* that you have a cold?

- Randomly sample 10000 assignments from  $p(x)$ .
- In 200 of them (2%), patient is sneezing with a fever and no cough.
- In 140 (1.4%) of those, the patient also has a cold.



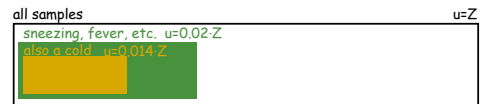
answer: 70% (0.014/0.02)

## Probabilistic interpretation → Predictions

You're sneezing with a fever & no cough.

Then what is the *probability* that you have a cold?

- Randomly sample 10000 assignments from  $p(x)$ .
- In 200 of them (2%), patient is sneezing with a fever and no cough.
- In 140 (1.4%) of those, the patient also has a cold.



answer: 70% (0.014-Z / 0.02-Z)

### Probabilistic interpretation → Predictions

You're sneezing with a fever & no cough.  
Then what is the *probability* that you have a cold?

- Randomly sample 10000 assignments from  $p(x)$ . *Could we compute exactly instead?*

*Remember, we can find this by variable elimination if necessary*

This too: just add unary constraints Sneezing=1, Fever=1, Cough=0

This too: one more unary constraint Cold=1

answer: 70% ( $0.014 \cdot Z / 0.02 \cdot Z$ )

600.325/425 Declarative Methods - J. Eisner 37

### Probabilistic interpretation → Learning

- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to **real data**)? **90% of the time**
- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to **the full model**)? **55% of the time**
  - I.e., if you randomly sample many assignments from  $p(x)$ , 55% of assignments have  $(1, 0, 1)$ .
  - E.g., 55% have (Cold, ~Cough, Sneeze): too few.
- To learn a better  $p(x)$ , we adjust the factor functions to bring the second ratio from 55% up to 90%.

600.325/425 Declarative Methods - J. Eisner 38

### Probabilistic interpretation → Learning

- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to real data)? **90% of the time**
- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to the full model)? **55% of the time**
- To learn a better  $p(x)$ , we adjust the factor functions to bring the second ratio from 55% up to 90%.

- By increasing  $f1(1, 0, 1)$ , we can increase the model's probability that  $(X1, X2, X3) = (1, 0, 1)$ .
- Unwanted ripple effect: This will also increase the model's probability that  $X3=1$ , and hence will change the probability that  $X5=1$ , and ...
- So we have to change all the factor functions at once to make all of them match real data.
- Theorem: This is always possible. (gradient descent or other algorithms)
  - Theorem: The resulting learned function  $p(x)$  maximizes  $p(\text{real data})$ .

600.325/425 Declarative Methods - J. Eisner 39

### Probabilistic interpretation → Learning

- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to real data)? **90% of the time**
- How likely is it for  $(X1, X2, X3) = (1, 0, 1)$  (according to the full model)? **55% of the time**
- To learn a better  $p(x)$ , we adjust the factor functions to bring the second ratio from 55% up to 90%.

- By increasing  $f1(1, 0, 1)$ , we can increase the model's probability that  $(X1, X2, X3) = (1, 0, 1)$ .
- Unwanted ripple effect: This will also increase the model's probability that  $X3=1$ , and hence will change the probability that  $X5=1$ , and ...
- So we have to change all the factor functions at once to make all of them match real data.
- Theorem: This is always possible. (gradient descent or other algorithms)
  - Theorem: The resulting learned function  $p(x)$  maximizes  $p(\text{real data})$ .

600.325/425 Declarative Methods - J. Eisner 40

### Probabilistic interpretation → Approximate constraint satisfaction

3. Central to deriving fast **approximation** algorithms.

- "Message passing" algorithms where nodes in the factor graph are repeatedly updated based on adjacent nodes.

- Gibbs sampling / simulated annealing
- Mean-field approximation and other variational methods
- Belief propagation
- Survey propagation

600.325/425 Declarative Methods - J. Eisner 41

### How do we sample from $p(x)$ ?

- Gibbs sampler: (should remind you of stochastic SAT solvers)
  - Pick a random starting assignment.
  - Repeat n times: Pick a variable and possibly flip it, at random
  - Theorem: Our new assignment is a random sample from a distribution close to  $p(x)$  (converges to  $p(x)$  as  $n \rightarrow \infty$ )

How do we decide whether new value should be 0 or 1?

If  $u(x)$  is twice as big when set at 0 than at 1, then pick 1 with prob  $2/3$ , pick 0 with prob  $1/3$ .

It's a local computation to determine that flipping the variable doubles  $u(x)$ , since only these factors of  $u(x)$  change.

600.325/425 Declarative Methods - J. Eisner 42

### Technique #3: Simulated annealing

- Gibbs sampler can sample from  $p(x)$ .
- Replace each factor  $f(x)$  with  $f(x)^\beta$ .
- Now  $p(x)$  is proportional to  $\prod f(x)^\beta$ , with  $\sum p(x) = 1$ .
- What happens as  $\beta \rightarrow \infty$ ?

- Sampler turns into a maximizer!
  - Let  $x^*$  be the value of  $x$  that maximizes  $p(x)$ .
  - For very large  $\beta$ , a single sample is almost always equal to  $x^*$ .
- Why doesn't this mean  $P=NP$ ?
  - As  $\beta \rightarrow \infty$ , need to let  $n \rightarrow \infty$  too to preserve quality of approx.
    - Sampler rarely goes down *steep* hills, so stays in local maxima for ages.
  - Hence, simulated annealing: gradually increase  $\beta$  as we flip variables.
  - Early on, we're flipping quite freely

600.325/425 Declarative Methods - J. Eisner 43

### Technique #4: Variational methods

- To work exactly with  $p(x)$ , we'd need to compute quantities like  $Z$ , which is NP-hard.
  - (e.g., to predict whether you have a cold, or to learn the factor functions)
- We saw that Gibbs sampling was a good (but slow) approximation that didn't require  $Z$ .
- The mean-field approximation is sort of like a deterministic "averaged" version of Gibbs sampling.
  - In Gibbs sampling, nodes flutter on and off – you can ask how often  $x_3$  was 1.
  - In mean-field approximation, every node maintains a belief about how often it's 1. This belief is updated based on the beliefs at adjacent nodes. No randomness.
  - [details beyond the scope of this course, but within reach]

600.325/425 Declarative Methods - J. Eisner 44

### Technique #4: Variational methods

- The mean-field approximation is sort of like a deterministic "averaged" version of Gibbs sampling.
  - In Gibbs sampling, nodes flutter on and off – you can ask how often  $x_3$  was 1.
  - In mean-field approximation, every node maintains a belief about how often it's 1. This belief is repeatedly updated based on the beliefs at adjacent nodes. No randomness.

600.325/425 Declarative Methods - J. Eisner 45

### Technique #4: Variational methods

- The mean-field approximation is sort of like a deterministic "averaged" version of Gibbs sampling.
  - Can frame this as seeking an optimal approximation of this  $p(x)$  ...

... by a  $p(x)$  defined as a product of simpler factors (easy to work with):

600.325/425 Declarative Methods - J. Eisner 46

### Technique #4: Variational methods

- More sophisticated version: Belief Propagation
  - The soft version of arc consistency
    - Arc consistency: some of my values become **impossible** → so do some of yours
    - Belief propagation: some of my values become **unlikely** → so do some of yours
      - Therefore, your other values become **more likely**
  - Note: Belief propagation has to be more careful than arc consistency about not having  $X$ 's influence on  $Y$  feed back and influence  $X$  as if it were separate evidence. Consider constraint  $X=Y$ .
    - But there will be feedback when there are cycles in the factor graph – which hopefully are long enough that the influence is not great. If no cycles (a tree), then the beliefs are exactly correct. In this case, BP boils down to a dynamic programming algorithm on the tree.
  - Can also regard it as Gibbs sampling without the randomness
    - That's what we said about mean-field, too, but this is an even better approx.
    - Gibbs sampling lets you see:
      - how often  $x_1$  takes each of its 2 values, 0 and 1.
      - how often  $(x_1, x_2, x_3)$  takes each of its 8 values such as  $(1, 0, 1)$ . (This is needed in learning if  $(x_1, x_2, x_3)$  is a factor.)
    - Belief propagation estimates these probabilities by "message passing."
    - Let's see how it works!

600.325/425 Declarative Methods - J. Eisner 47

### Technique #4: Variational methods

- Mean-field approximation
- Belief propagation
- Survey propagation:
  - Like belief propagation, but also assess the belief that the value of this variable doesn't matter! Useful for solving hard random 3-SAT problems.
- Generalized belief propagation: Joins constraints, roughly speaking.
- Expectation propagation: More approximation when belief propagation runs too slowly.
- Tree-reweighted belief propagation: ...

600.325/425 Declarative Methods - J. Eisner 48

### Great Ideas in ML: Message Passing

*Count the soldiers*

adapted from MacKay (2003) textbook

49

### Great Ideas in ML: Message Passing

*Count the soldiers*

adapted from MacKay (2003) textbook

50

### Great Ideas in ML: Message Passing

*Count the soldiers*

adapted from MacKay (2003) textbook

51

### Great Ideas in ML: Message Passing

*Each soldier receives reports from all branches of tree*

adapted from MacKay (2003) textbook

52

### Great Ideas in ML: Message Passing

*Each soldier receives reports from all branches of tree*

adapted from MacKay (2003) textbook

53

### Great Ideas in ML: Message Passing

*Each soldier receives reports from all branches of tree*

adapted from MacKay (2003) textbook

54

### Great Ideas in ML: Message Passing

Each soldier receives reports from all branches of tree

adapted from MacKay (2003) textbook

### Great Ideas in ML: Message Passing

Each soldier receives reports from all branches of tree

adapted from MacKay (2003) textbook

### Great ideas in ML: Belief Propagation

- In the CRF, message passing = forward-backward

57

### Great ideas in ML: Loopy Belief Propagation

- Extend CRF to "skip chain" to capture non-local factor
  - More influences on belief ☹️

58

### Great ideas in ML: Loopy Belief Propagation

- Extend CRF to "skip chain" to capture non-local factor
  - More influences on belief ☹️
  - Graph becomes loopy ☹️

Red messages not independent? Pretend they are!

59

### Technique #4: Variational methods

- Mean-field approximation
- Belief propagation
- Survey propagation:
  - Like belief propagation, but also assess the belief that the value of this variable doesn't matter! Useful for solving hard random 3-SAT problems.
- Generalized belief propagation: Joins constraints, roughly speaking.
- Expectation propagation: More approximation when belief propagation runs too slowly.
- Tree-reweighted belief propagation: ...

600.325 / 625  
 Declarative  
 Method  
 5/8  
 Eisner