BREGMAN CONDITIONAL RANDOM FIELDS:
SEQUENCE LABELING WITH PARALLELIZABLE INFERENCE ALGORITHMS

Caio Corro[1], Mathieu Lacroix[2], Joseph Le Roux[2]

[1]INSA Rennes, IRISA, Inria, CNRS, Université de Rennes, France
[2]Université Sorbonne Paris Nord, CNRS, LIPN, France

SEQUENCE LABELING

Problem

Given an input sequence, predict one output per element of the sequence, for example one tag per word of an input sentence.

- Part-of-speech tagging

PRP	**VB**	**DET**	**NN**
They	walk	the	dog

- Flat named-entity recognition with BIO tags

B-Per	**I-Per**	**O**	**O**	**B-Loc**
Neil	Armstrong	visited	the	moon

- Joint word segmentation and part-of-speech tagging with BIES tags

B-NN	**E-NN**	**S-JJ**	**B-CD**	**E-CD**	**B-NNB**	**E-NNB**	**S-,**	**S-VC**	**B-NNP**	**I-NNP**	**E-NNP**	**B-JJ**	**E-JJ**	**B-NN**	**E-NN**	**S-DEC**	**S-CD**	**S-.**
乐	章	长	廿	五	分	钟	，	为	贝,	多	芬	最	长	乐	章	之	一	。

SEQUENCE LABELING

Problem

Given an input sequence, predict one output per element of the sequence, for example one tag per word of an input sentence.

- Part-of-speech tagging

PRP	**VB**	**DET**	**NN**
They	walk	the	dog

- Flat named-entity recognition with BIO tags

B-Per	**I-Per**	**O**	**O**	**B-Loc**
Neil	Armstrong	visited	the	moon

- Joint word segmentation and part-of-speech tagging with BIES tags

B-NN	**E-NN**	**S-JJ**	**B-CD**	**E-CD**	**B-NNB**	**E-NNB**	**S-,**	**S-VC**	**B-NNP**	**I-NNP**	**E-NNP**	**B-JJ**	**E-JJ**	**B-NN**	**E-NN**	**S-DEC**	**S-CD**	**S-.**
乐	章	长	廿	五	分	钟	，	为	贝	多	芬	最	长	乐	章	之	一	。

GRAPH-BASED DECODING

Tags as Vertices

For each word, create one vertex per tag where vertex weights are neural network outputs.

Prediction / Decoding

Select on vertex per word.

In practice we have way more tags

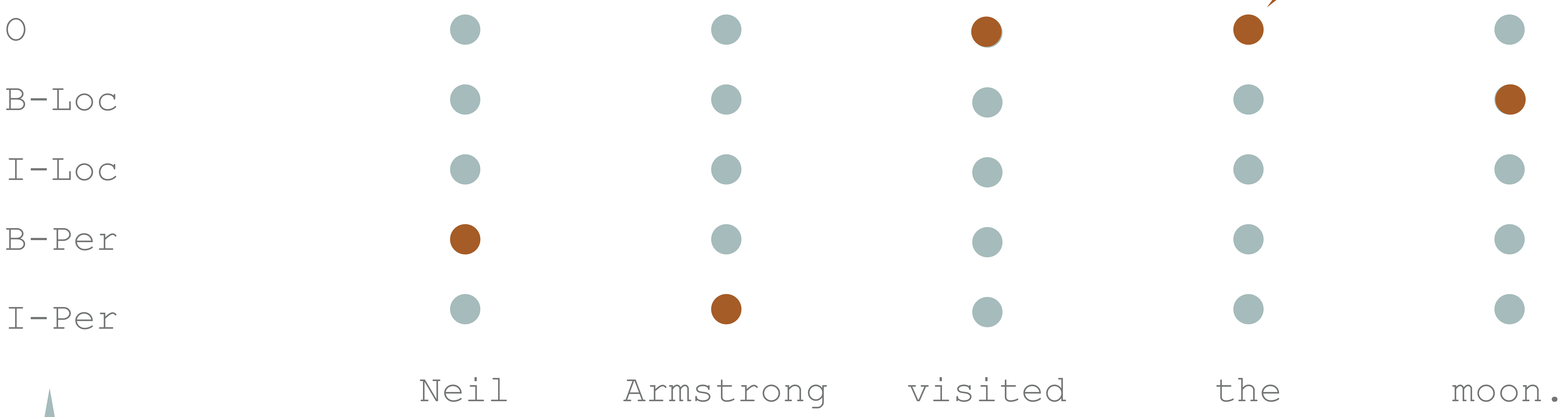
GRAPH-BASED DECODING

Tags as Vertices

For each word, create one vertex per tag where vertex weights are neural network outputs.

Prediction / Decoding

Select on vertex per word.



Red vertices are the best vertex for each word

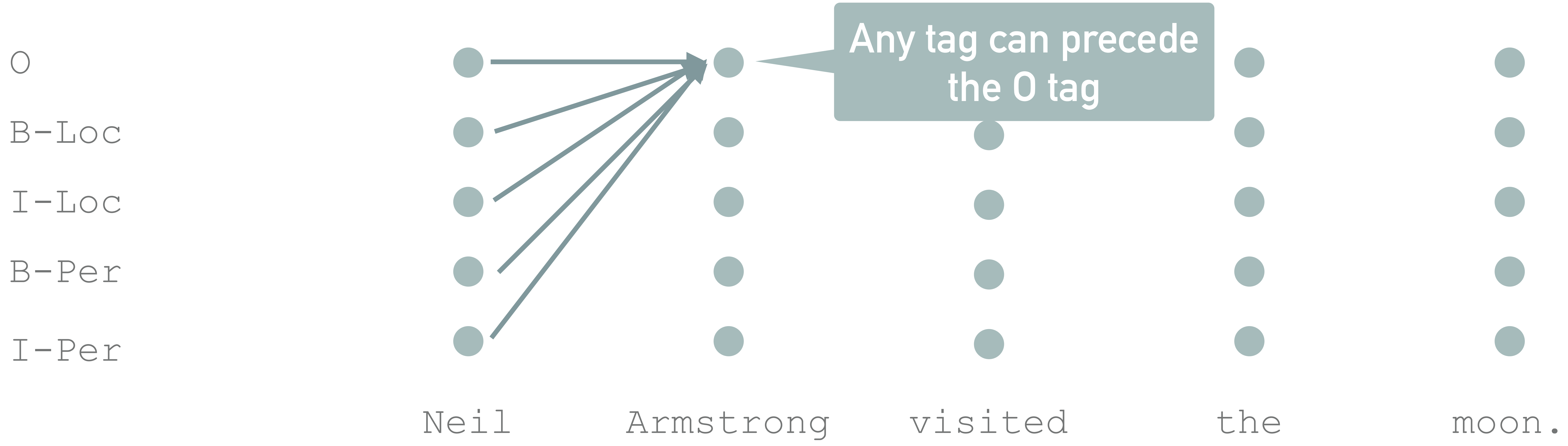
In practice we have way more tags

GRAPH-BASED DECODING

Transitions as Arcs

Add arcs between adjacent vertices:

- arc weights are neural network outputs
- do not introduce arcs for forbidden tag transitions (or set its weight to -∞)



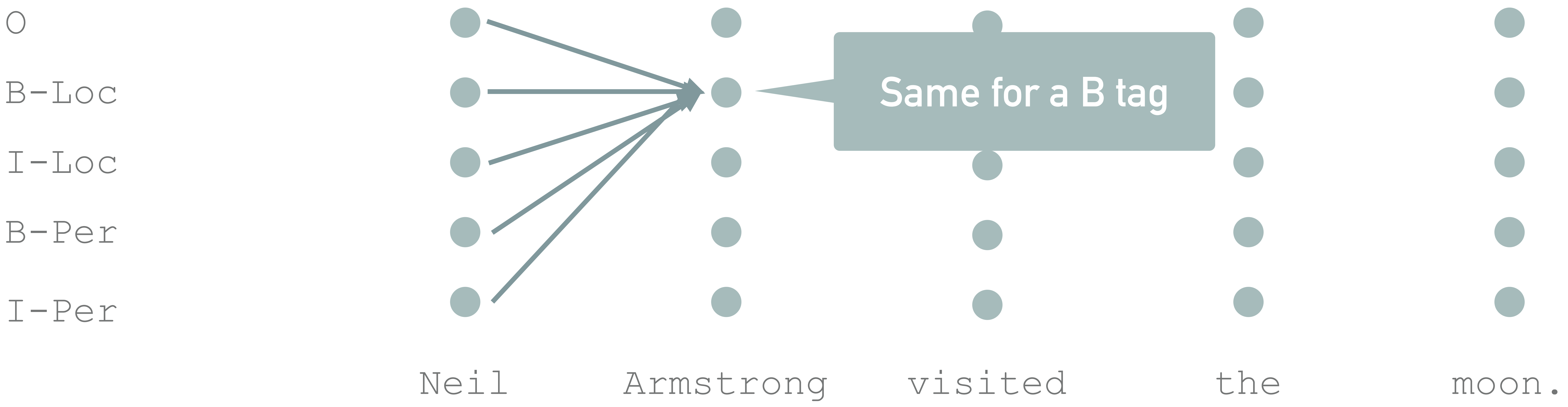
Any tag can precede the O tag

GRAPH-BASED DECODING

Transitions as Arcs

Add arcs between adjacent vertices:

- arc weights are neural network outputs
- do not introduce arcs for forbidden tag transitions (or set its weight to -∞)

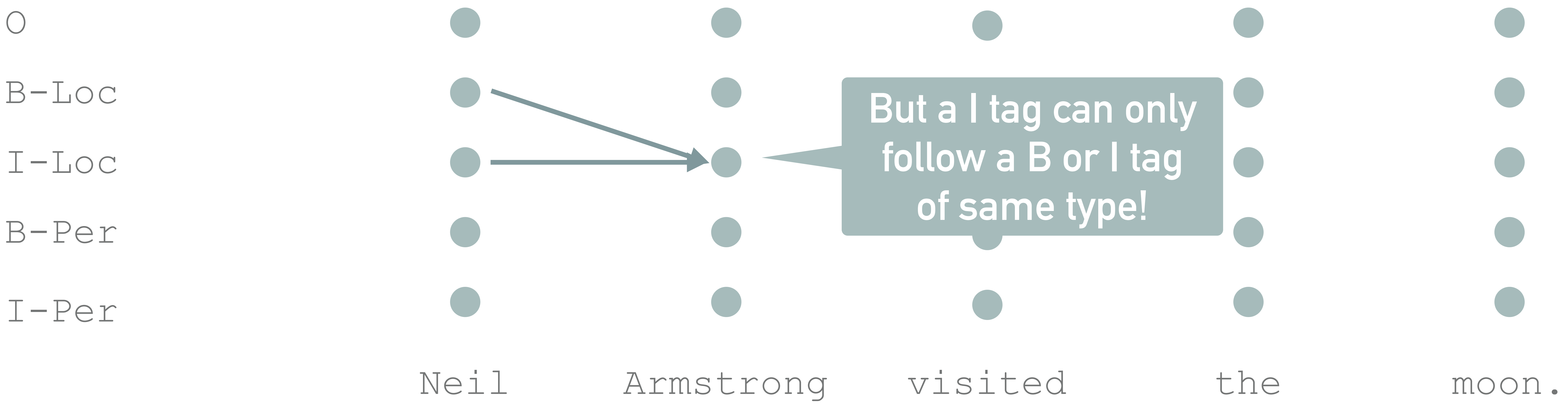


GRAPH-BASED DECODING

Transitions as Arcs

Add arcs between adjacent vertices:

- arc weights are neural network outputs
- do not introduce arcs for forbidden tag transitions (or set its weight to -∞)



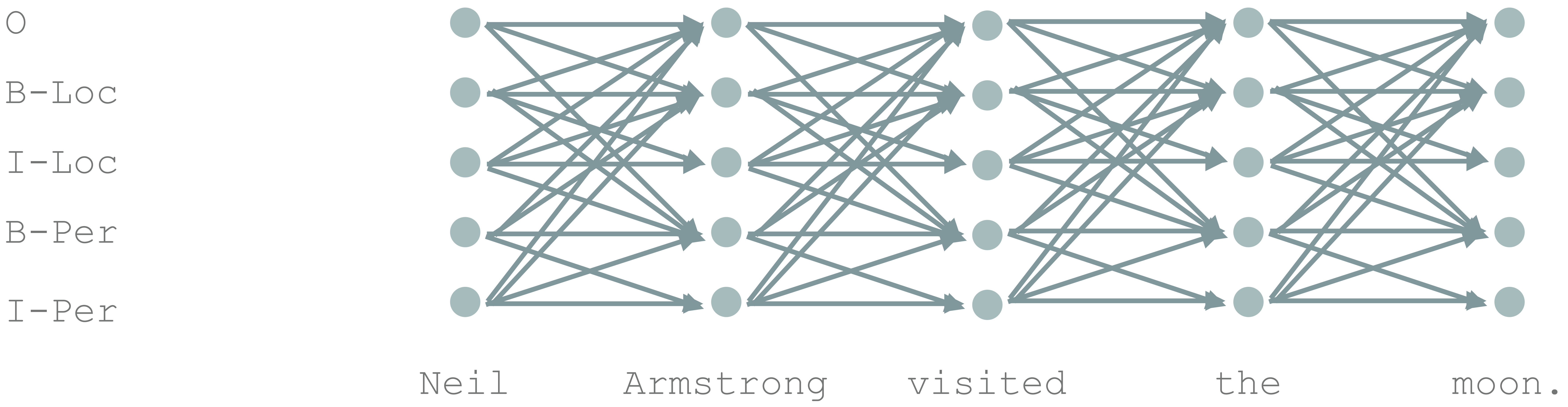
But a I tag can only follow a B or I tag of same type!

GRAPH-BASED DECODING

Transitions as Arcs

Add arcs between adjacent vertices:

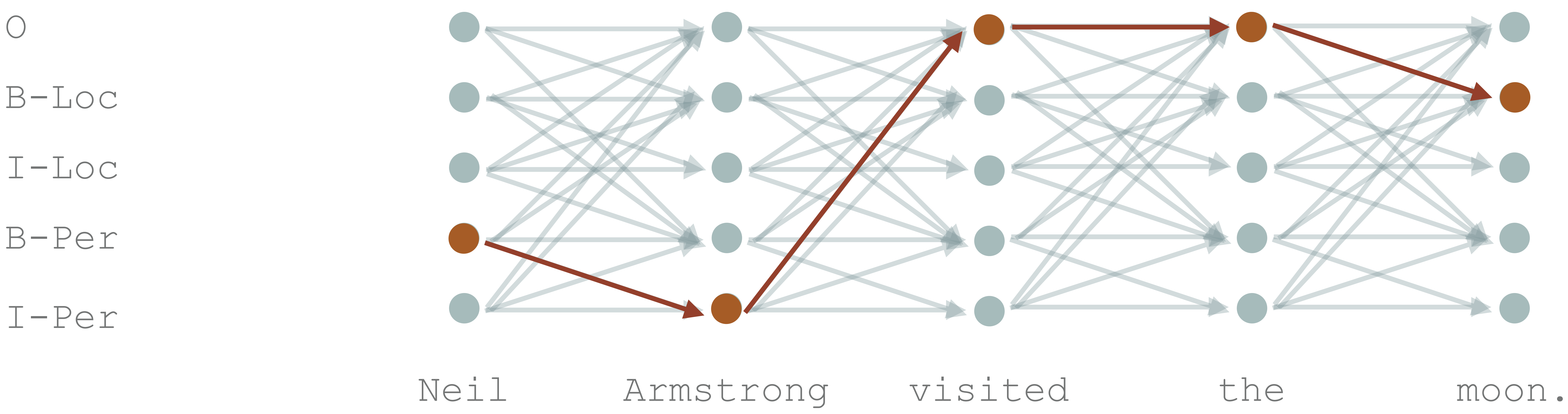
- arc weights are neural network outputs
- do not introduce arcs for forbidden tag transitions (or set its weight to -∞)



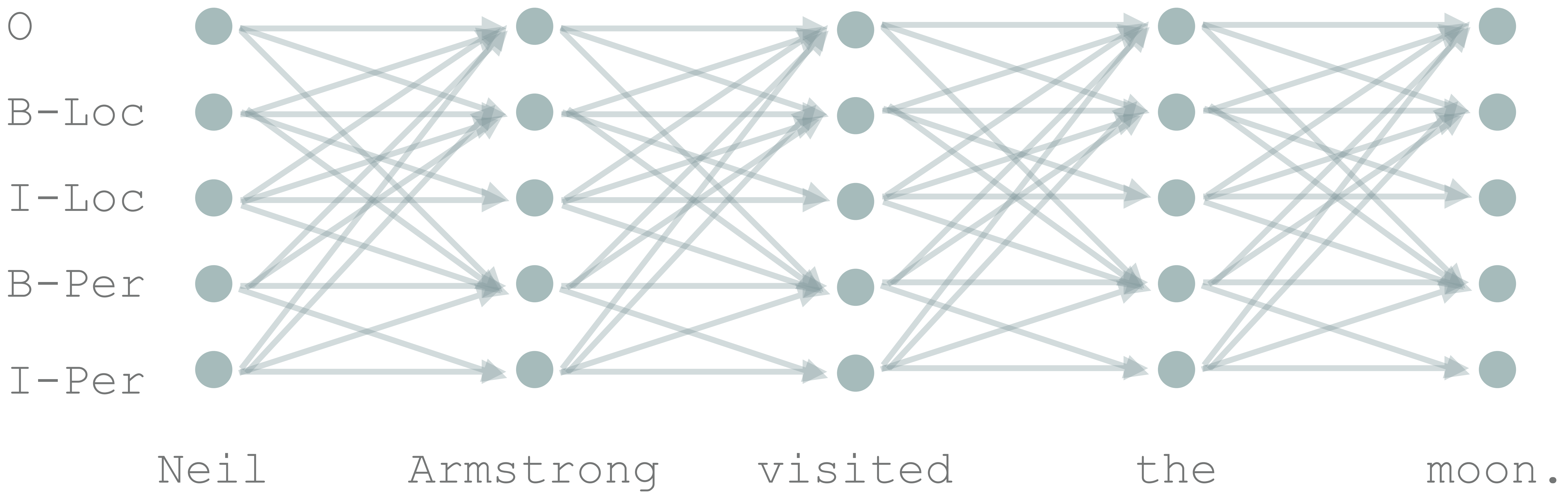
GRAPH-BASED DECODING

Sequence Labelings as Paths in the Viterbi Trellis

- A path from the source vertex to the target vertex represent a tagged sentence (1-to-1 correspondance)
- The prediction of the model is the path of maximum weight



GRAPH-BASED DECODING



O

B-Loc

I-Loc

B-Per

I-Per

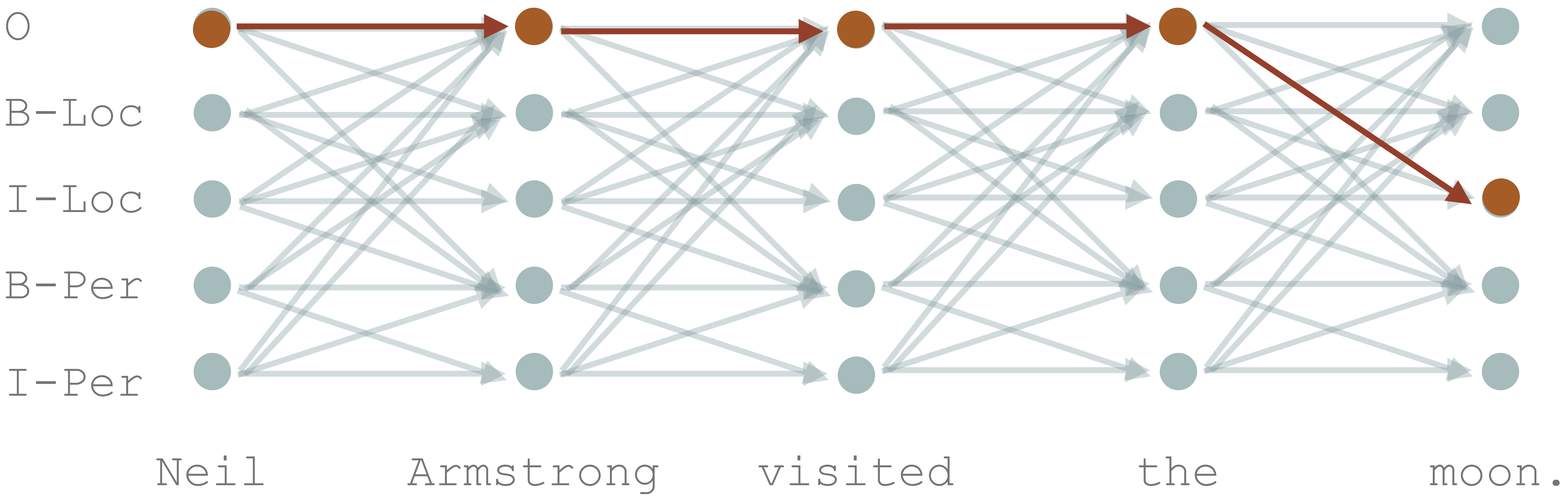
Neil Armstrong visited the moon.

Path Weighting

- Transition weight vector: (given by the neural net)

$$\mathbf{w}^\top = [\ \texttt{+4.23}\ \ \texttt{-3.16}\ \ ..\ \ \texttt{+1.02}\ \ ..\ \ \texttt{+5.36}\ \ ..\ \ \texttt{+0.46}\ \ ..\ \ \texttt{-3.67}\ \ \texttt{+0.60}\ \ \texttt{-1.64}\ ]$$

GRAPH-BASED DECODING



Path Weighting

- Transition weight vector: (given by the neural net)
- Arc selection vectors: ({0,1} vectors)

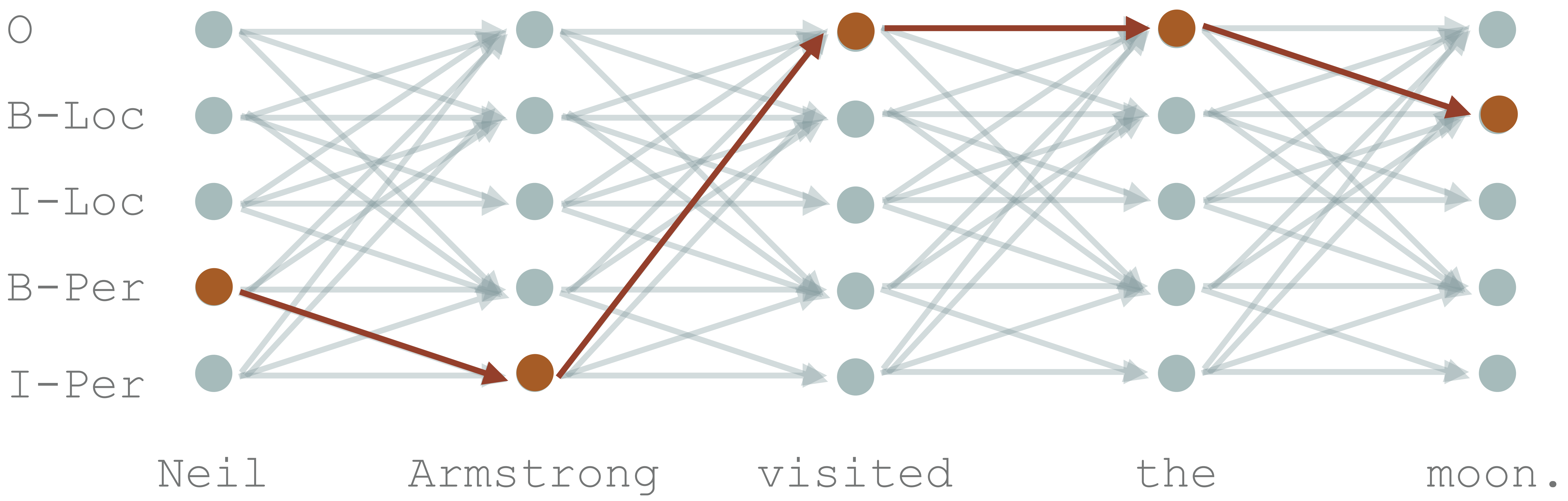
$$\mathbf{w}^\top = [\ +4.23\quad -3.16\quad ..\quad +1.02\quad ..\quad +5.36\quad ..\quad +0.46\quad ..\quad -3.67\quad +0.60\quad -1.64\ ]$$

$$\mathbf{q}_1^\top = [\ 1\quad 0\quad ..\quad 1\quad ..\quad 1\quad ..\quad 1\quad ..\quad 0\quad 0\quad 0\ ]$$

Not all binary vectors are valid paths!

The weight of a path is the inner product between the two vectors: $\langle \mathbf{w}, \mathbf{q} \rangle$

GRAPH-BASED DECODING



Path Weighting

- Transition weight vector: (given by the neural net)
- Arc selection vectors: ({0,1} vectors)

Not all binary vectors are valid paths!

$$\mathbf{w}^\top = [\ +4.23 \quad -3.16 \quad .. \quad +1.02 \quad .. \quad +5.36 \quad .. \quad +0.46 \quad .. \quad -3.67 \quad +0.60 \quad -1.64\ ]$$

$$\mathbf{q}_1^\top = [\ 1 \quad 0 \quad .. \quad 1 \quad .. \quad 1 \quad .. \quad 1 \quad .. \quad 0 \quad 0 \quad 0\ ]$$

$$\mathbf{q}_2^\top = [\ 0 \quad 1 \quad .. \quad 0 \quad .. \quad 1 \quad .. \quad 0 \quad .. \quad 0 \quad 1 \quad 0\ ]$$

etc.

The weight of a path is the inner product between the two vectors: $\langle \mathbf{w}, \mathbf{q} \rangle$

CONDITIONAL RANDOM FIELDS

$$\begin{bmatrix} +4.23 \\ -3.16 \\ \vdots \\ +1.02 \\ \vdots \\ +5.36 \\ \vdots \\ +0.46 \\ \vdots \\ -3.67 \\ +0.60 \\ -1.64 \end{bmatrix} \mathbf{w}$$

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis.

Then $\mathbf{M}^\top\mathbf{w}$ is vector containing the weight of each path.

$\mathbf{M}^\top$

$\mathbf{M}^\top\mathbf{w}$

CONDITIONAL RANDOM FIELDS

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis.
Then $\mathbf{M}^\top\mathbf{w}$ is vector containing the weight of each path.

$$\begin{bmatrix} +4.23 \\ -3.16 \\ \vdots \\ +1.02 \\ \vdots \\ +5.36 \\ \vdots \\ +0.46 \\ \vdots \\ -3.67 \\ +0.60 \\ -1.64 \end{bmatrix} \mathbf{w}$$

q₁

Weight of q₁

$$\mathbf{M}^\top \begin{bmatrix} 1 & 0 & .. & 1 & .. & 1 & .. & 1 & .. & 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} +11.07 \end{bmatrix} \mathbf{M}^\top\mathbf{w}$$

CONDITIONAL RANDOM FIELDS

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis.
Then $\mathbf{M}^\top\mathbf{w}$ is vector containing the weight of each path.

$$\begin{bmatrix} +4.23 \\ -3.16 \\ \vdots \\ +1.02 \\ \vdots \\ +5.36 \\ \vdots \\ +0.46 \\ \vdots \\ -3.67 \\ +0.60 \\ -1.64 \end{bmatrix} \ \mathbf{w}$$

q_1

$$\mathbf{M}^\top \begin{bmatrix} 1 & 0 & .. & 1 & .. & 1 & .. & 1 & .. & 0 & 0 & 0 \\ 0 & 1 & .. & 0 & .. & 1 & .. & 0 & .. & 0 & 1 & 0 \\ & & & & & & & & & & & \end{bmatrix} \qquad \begin{bmatrix} +11.07 \\ +3.68 \\ \\ \end{bmatrix} \ \mathbf{M}^\top\mathbf{w}$$

q_2

Weight of q_1

Weight of q_2

CONDITIONAL RANDOM FIELDS

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis.
Then $\mathbf{M}^\top \mathbf{w}$ is vector containing the weight of each path.

$$\mathbf{w} = \begin{bmatrix} +4.23 \\ -3.16 \\ \vdots \\ +1.02 \\ \vdots \\ +5.36 \\ \vdots \\ +0.46 \\ \vdots \\ -3.67 \\ +0.60 \\ -1.64 \end{bmatrix}$$

$$\mathbf{M}^\top \quad \begin{array}{c} q_1 \\ q_2 \\ \\ \end{array} \begin{bmatrix} 1 & 0 & .. & 1 & .. & 1 & .. & 1 & .. & 0 & 0 & 0 \\ 0 & 1 & .. & 0 & .. & 1 & .. & 0 & .. & 0 & 1 & 0 \\ & & & & & \textit{etc etc etc.} & & & & & & \end{bmatrix}$$

$$\mathbf{M}^\top \mathbf{w} = \begin{array}{c} \text{Weight of } q_1 \\ \text{Weight of } q_2 \\ \\ \end{array} \begin{bmatrix} +11.07 \\ +3.68 \\ \textit{etc.} \end{bmatrix}$$

CONDITIONAL RANDOM FIELDS

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis.
Then $\mathbf{M}^\top \mathbf{w}$ is vector containing the weight of each path.

Conditional Random Fields (CRF)

Distribution over sequence labelings defined as:

$$p_\theta(\mathbf{q} \,|\, \mathbf{s}) = \exp\big(\, \langle \mathbf{q}, f_\theta(\mathbf{s}) \rangle \;-\; A_Y(f_\theta(\mathbf{s}))\, \big)$$

Softmax over structures

fθ is the neural net parameterized by θ

where the log-partition ensures that the distribution is well-defined:

$$A_Y(\mathbf{w}) = \log \sum_i \exp\left[\mathbf{M}^\top \mathbf{w}\right]_i$$

CONDITIONAL RANDOM FIELDS

Structure Encoding Matrix

Let M be a matrix s.t. each column encodes one path in the trellis. Then $\mathbf{M}^\top \mathbf{w}$ is vector containing the weight of each path.

Conditional Random Fields (CRF)

Distribution over sequence labelings defined as:

$$p_\theta(\mathbf{q} \,|\, \mathbf{s}) = \exp\Big(\langle \mathbf{q}, f_\theta(\mathbf{s}) \rangle \; - \; A_Y(f_\theta(\mathbf{s})) \Big)$$

Softmax over structures

f_θ is the neural net parameterized by θ

where the log-partition ensures that the distribution is well-defined:

$$A_Y(\mathbf{w}) = \log \sum_i \exp \left[\mathbf{M}^\top \mathbf{w}\right]_i$$

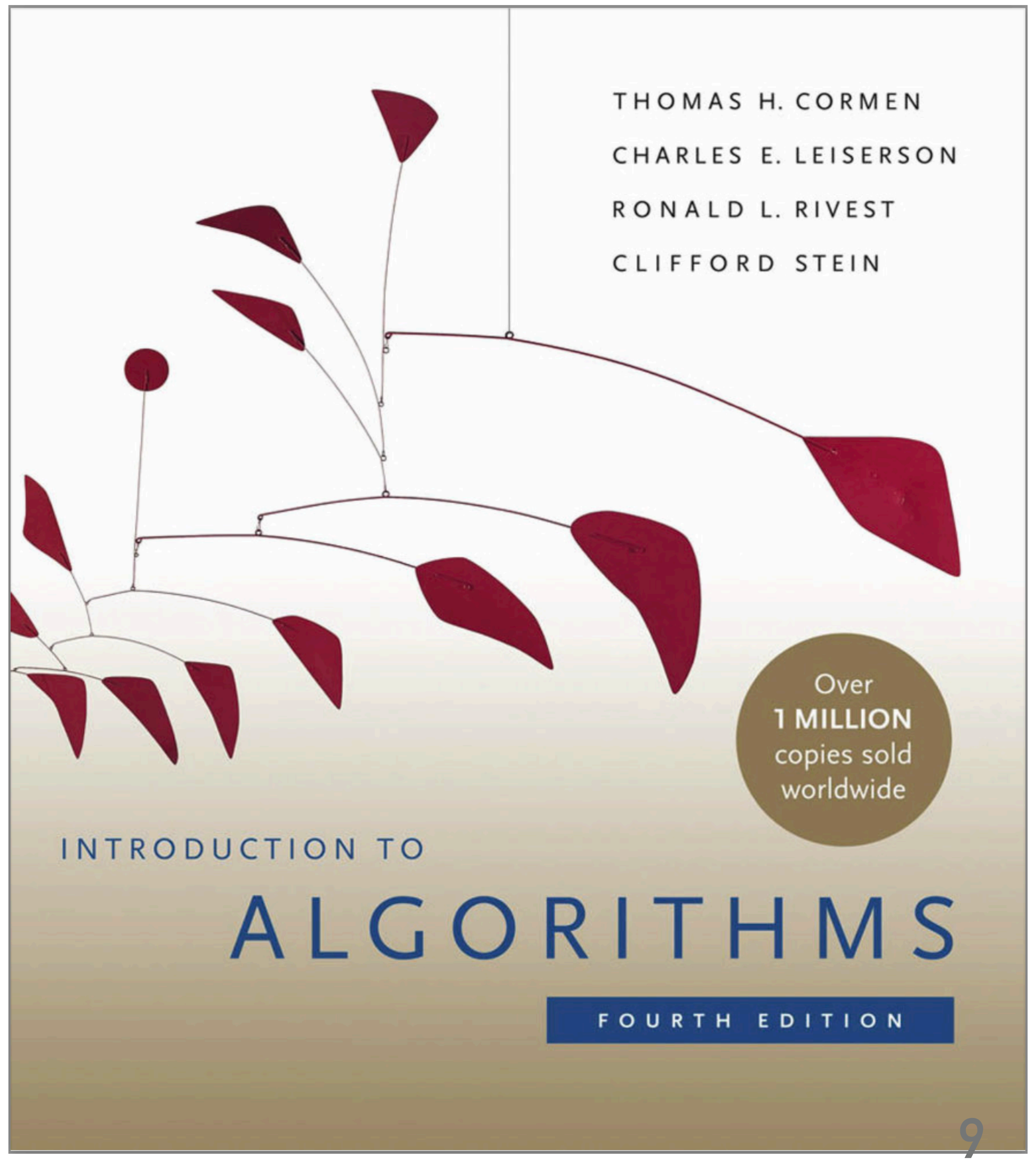
Inference Problems

- MAP inference:
  compute the best sequence of tags (for prediction)
- Marginal inference:
  compute the log-partition function (for training, NLL loss)

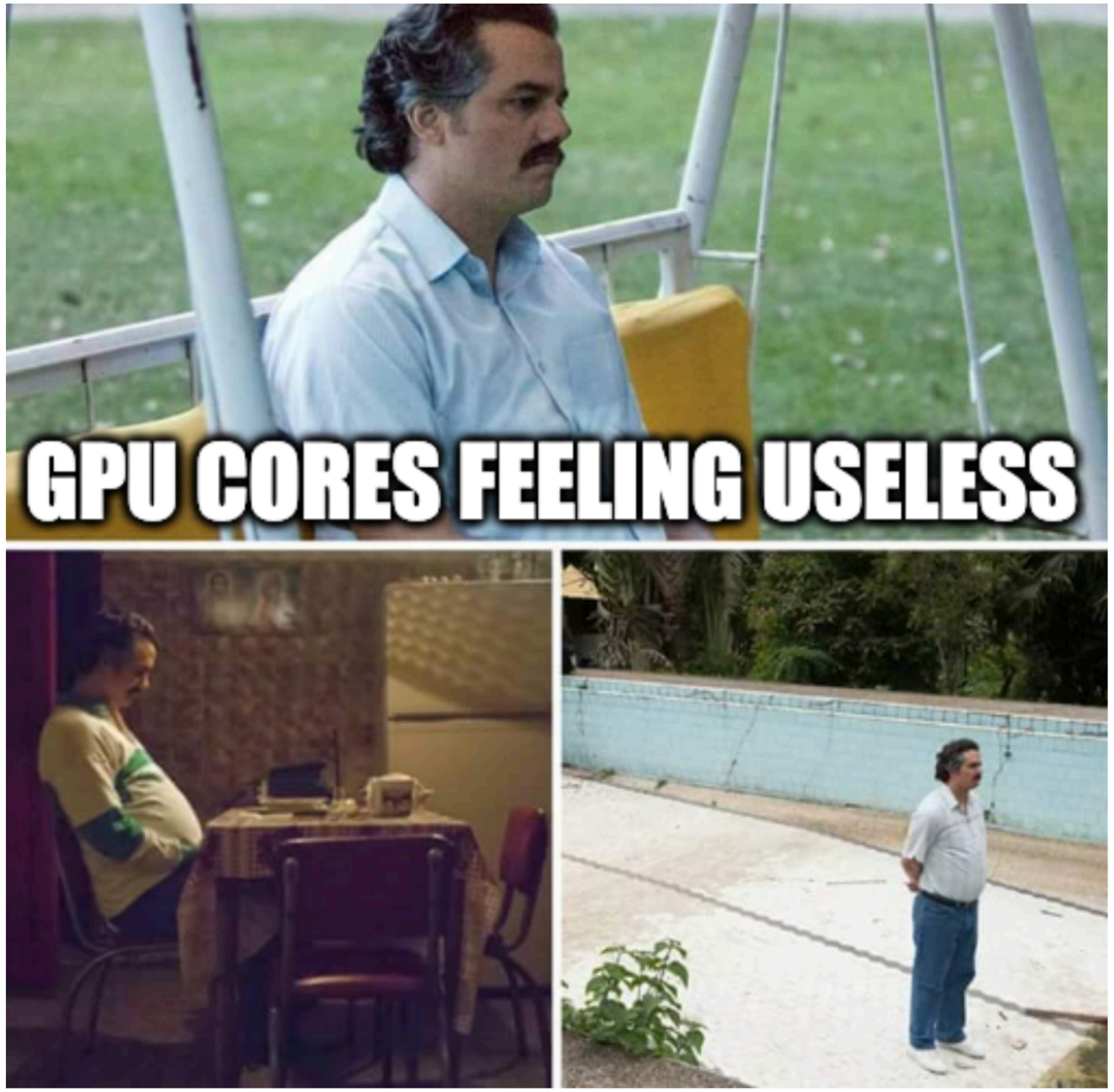
Inference Algorithms

Via dynamic programming:

- Viterbi
- Forward



CONDITIONAL RANDOM FIELDS



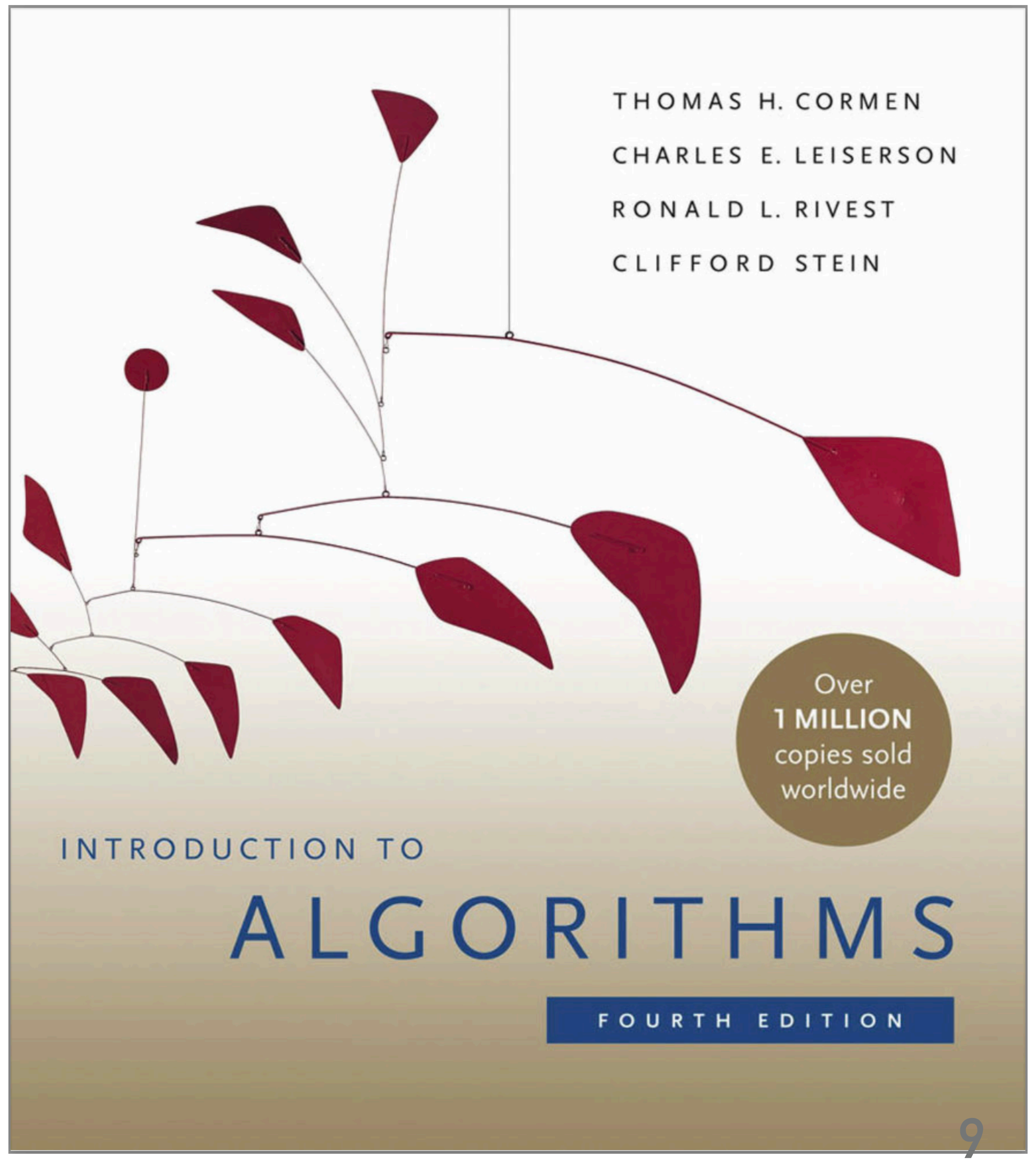
Inference Problems

- MAP inference:
  compute the best sequence of tags (for prediction)
- Marginal inference:
  compute the log-partition function (for training, NLL loss)

Inference Algorithms

Via dynamic programming:

- Viterbi
- Forward



These CRF algorithms cannot fully leverage parallelization capabilities of GPUs!!!

BREGMAN CONDITIONAL RANDOM FIELDS

Conditional Random Fields (CRF)

The log-partition function of a CRF whose structure is encoded by matrix M is defined as follows:

$$\begin{aligned} A_Y(\mathbf{w}) &= \log \sum_i \exp \left[\mathbf{M}^\top \mathbf{w}\right]_i \\ &= \max_{\mathbf{p} \in \triangle_Y} \langle \mathbf{p}, \mathbf{M}^\top \mathbf{w} \rangle + H(\mathbf{p}) \end{aligned}$$

Distribution regularization via Shannon entropy

BREGMAN CONDITIONAL RANDOM FIELDS

Conditional Random Fields (CRF)

The log-partition function of a CRF whose structure is encoded by matrix M is defined as follows:

$$\begin{aligned} A_Y(\mathbf{w}) &= \log \sum_i \exp \left[\mathbf{M}^\top \mathbf{w}\right]_i \\ &= \max_{\mathbf{p} \in \triangle_Y} \langle \mathbf{p}, \mathbf{M}^\top \mathbf{w} \rangle + H(\mathbf{p}) \end{aligned}$$

Distribution regularization
via Shannon entropy

Setting $\mathbf{q} = \mathbf{M}\mathbf{p}$ and optimizing over $\mathbf{q} \in \left\{\mathbf{M}\mathbf{p} \,|\, \mathbf{p} \in \triangle_Y\right\} = \mathbf{conv}\; Y$

we obtain:

$$= \max_{\mathbf{q} \in \mathbf{conv}\, Y} \langle \mathbf{q}, \mathbf{w} \rangle - \boxed{R(\mathbf{q})}$$

Defined so equality holds

BREGMAN CONDITIONAL RANDOM FIELDS

Conditional Random Fields (CRF)

The log-partition function of a CRF whose structure is encoded by matrix M is defined as follows:

$$A_Y(\mathbf{w}) = \log \sum_i \exp \left[\mathbf{M}^\top \mathbf{w}\right]_i$$

$$= \max_{\mathbf{p}\in\triangle_Y} \langle \mathbf{p}, \mathbf{M}^\top \mathbf{w}\rangle + H(\mathbf{p})$$

Distribution regularization via Shannon entropy

Setting $\mathbf{q} = \mathbf{Mp}$ and optimizing over $\mathbf{q} \in \left\{\mathbf{Mp} \,|\, \mathbf{p} \in \triangle_Y\right\} = \mathbf{conv}\, Y$ we obtain:

$$= \max_{\mathbf{q}\in\mathbf{conv}\,Y} \langle \mathbf{q}, \mathbf{w}\rangle - \boxed{R(\mathbf{q})}$$

Defined so equality holds

Bregman CRF

A Bregman CRF defines a probability distribution over sequence labeling whose marginal distribution is defined by:

$$B_Y(\mathbf{w}) = \max_{\mathbf{p}\in\triangle_Y} \langle \mathbf{p}, \mathbf{M}^\top \mathbf{w}\rangle + H(\mathbf{Mp})$$

Using the same change of variable, we obtain:

$$= \max_{\mathbf{q}\in\mathbf{conv}\,Y} \langle \mathbf{q}, \mathbf{w}\rangle + \boxed{H(\mathbf{q})}$$

Mean regularization

Benefits

- $\mathbf{q}$ is of polynomial size
- can be rewritten as a KL projection!
- both approximate MAP and marginal inference reduce to the same algorithm

$$\operatorname*{argmin}_{\mathbf{q}\in\mathbf{conv}\,Y} D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

BREGMAN CONDITIONAL RANDOM FIELDS

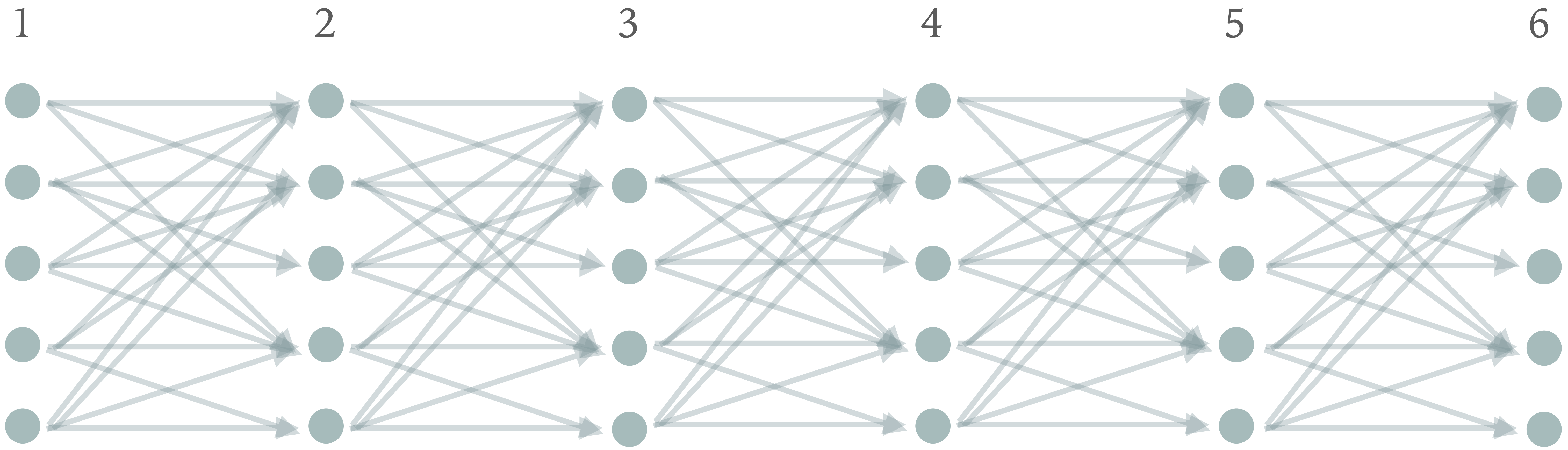
Optimization Problem

We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\mathbf{conv}\,Y}{\operatorname{argmin}}\; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \underset{\mathbf{q}\in C_1\cap C_2}{\operatorname{argmin}}\; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.



BREGMAN CONDITIONAL RANDOM FIELDS

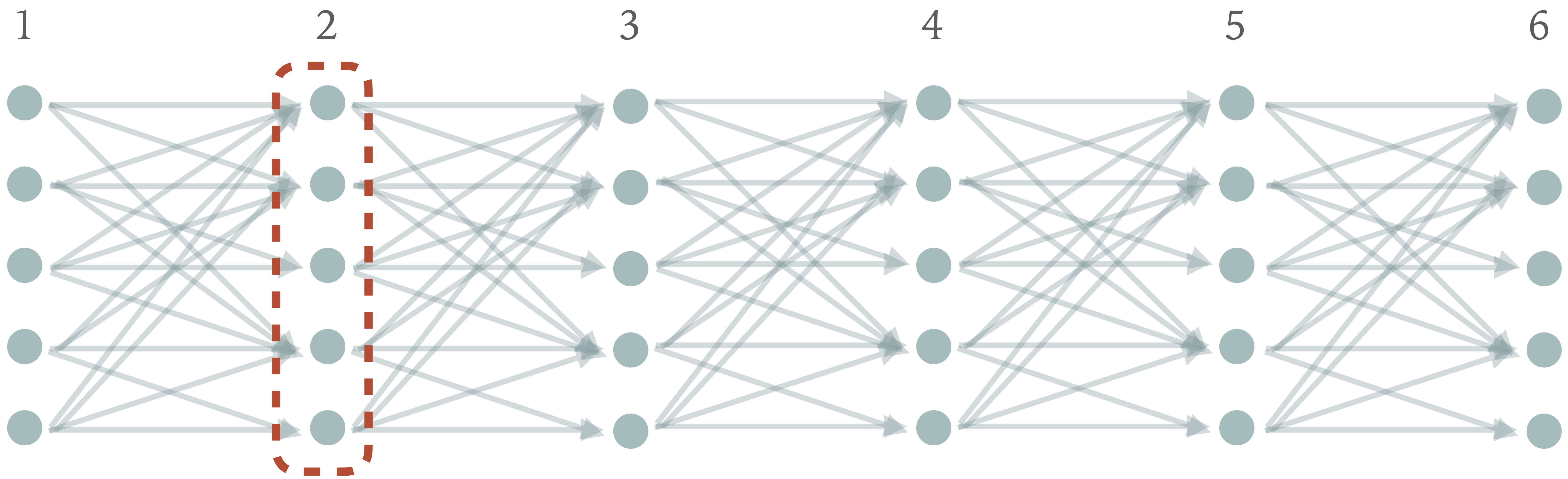
Optimization Problem

We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q} \in \mathbf{conv}\, Y}{\operatorname{argmin}} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \underset{\mathbf{q} \in C_1 \cap C_2}{\operatorname{argmin}} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.



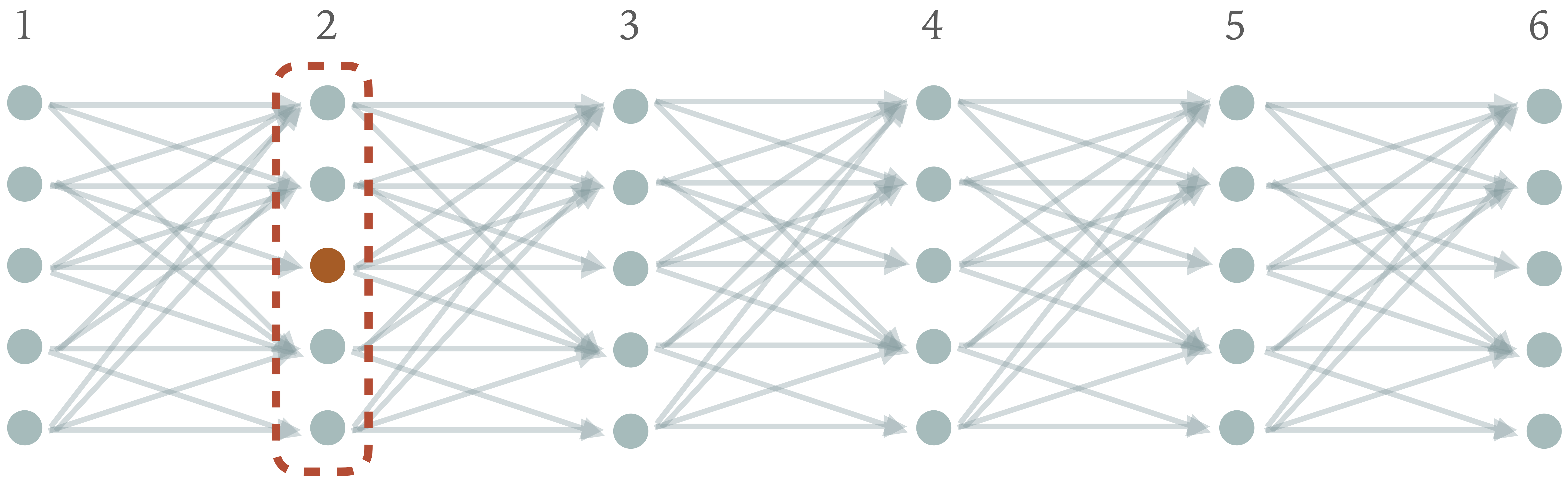
Constraints

At a given position, the following constraints must hold:

- Exactly one vertex is selected
- This vertex has exactly one incoming and one outgoing arc

BREGMAN CONDITIONAL RANDOM FIELDS

Optimization Problem



We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\mathbf{conv}\,Y}{\operatorname{argmin}} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \underset{\mathbf{q}\in C_1 \cap C_2}{\operatorname{argmin}} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.

Constraints

At a given position, the following constraints must hold:

- Exactly one vertex is selected
- This vertex has exactly one incoming and one outgoing arc

BREGMAN CONDITIONAL RANDOM FIELDS

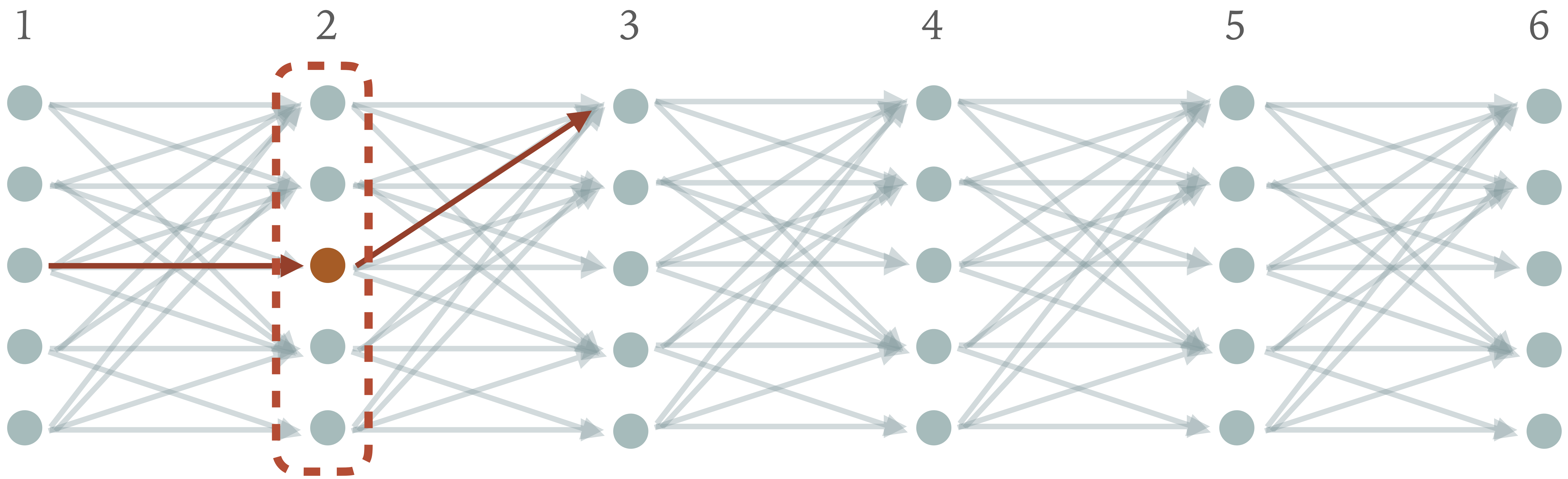
Optimization Problem

We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\mathbf{conv}\,Y}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\ \|\ \exp\mathbf{w})$$

$$= \underset{\mathbf{q}\in C_1\cap C_2}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\ \|\ \exp\mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.



Constraints

At a given position, the following constraints must hold:

- Exactly one vertex is selected
- This vertex has exactly one incoming and one outgoing arc

BREGMAN CONDITIONAL RANDOM FIELDS

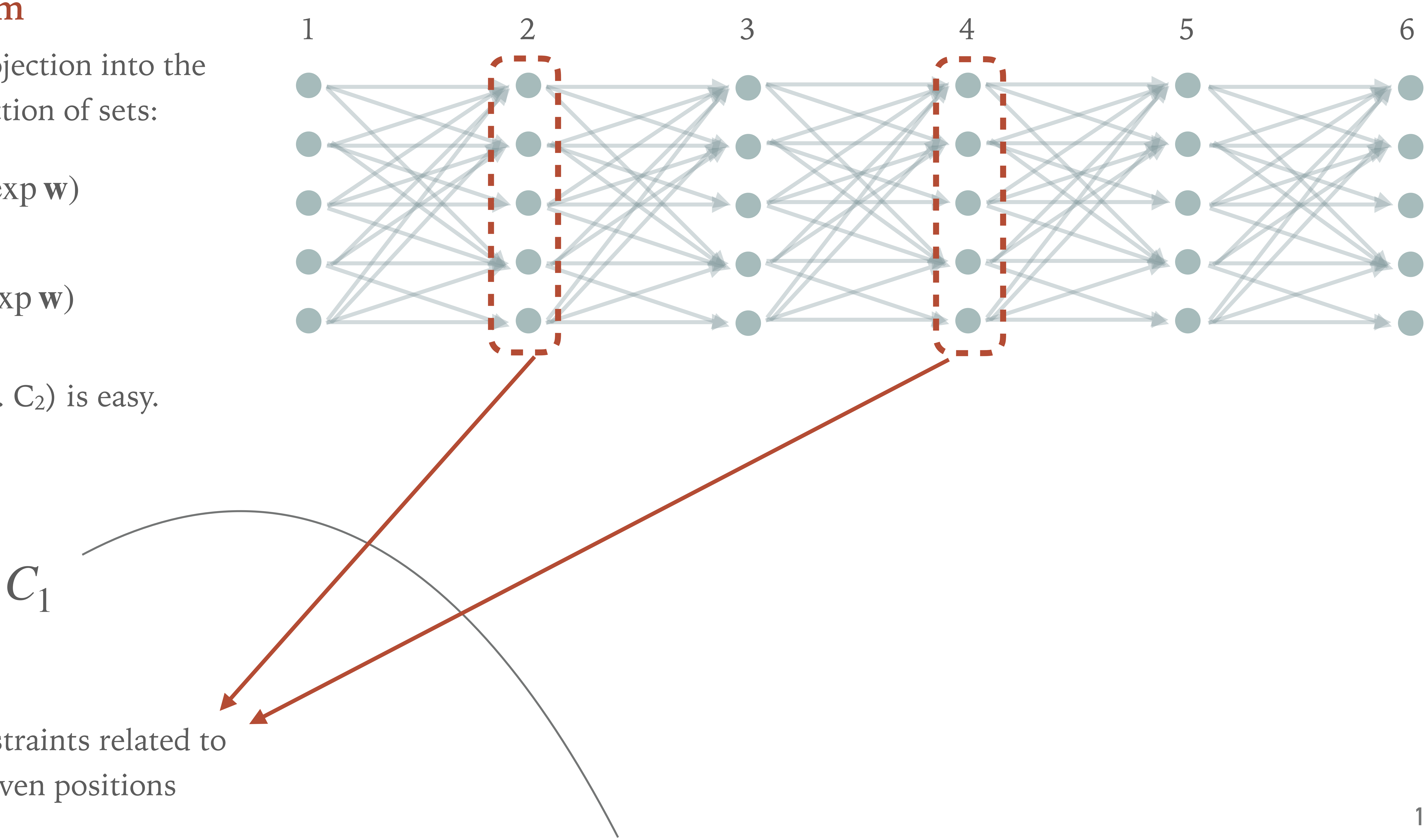
Optimization Problem

We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q}\in\mathbf{conv}\,Y} D_{KL}(\mathbf{q}\,\|\,\exp\mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q}\in C_1\cap C_2} D_{KL}(\mathbf{q}\,\|\,\exp\mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.



Constraints related to even positions

BREGMAN CONDITIONAL RANDOM FIELDS

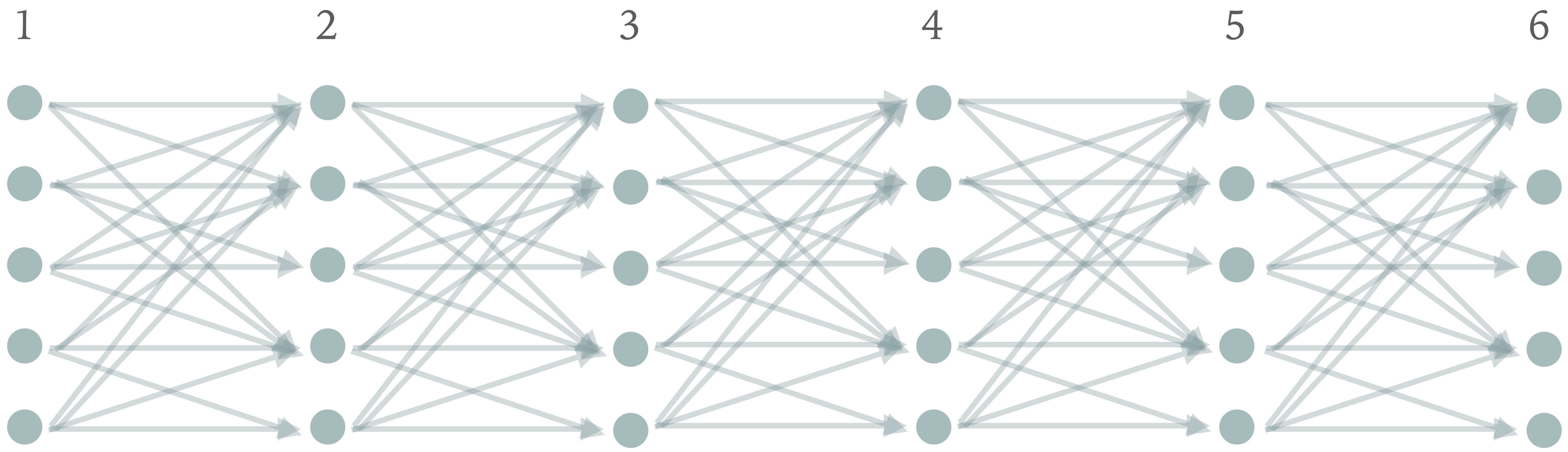
Optimization Problem

We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q}\in\mathbf{conv}\,Y} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q}\in C_1\cap C_2} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.



Constraints related to even positions

BREGMAN CONDITIONAL RANDOM FIELDS

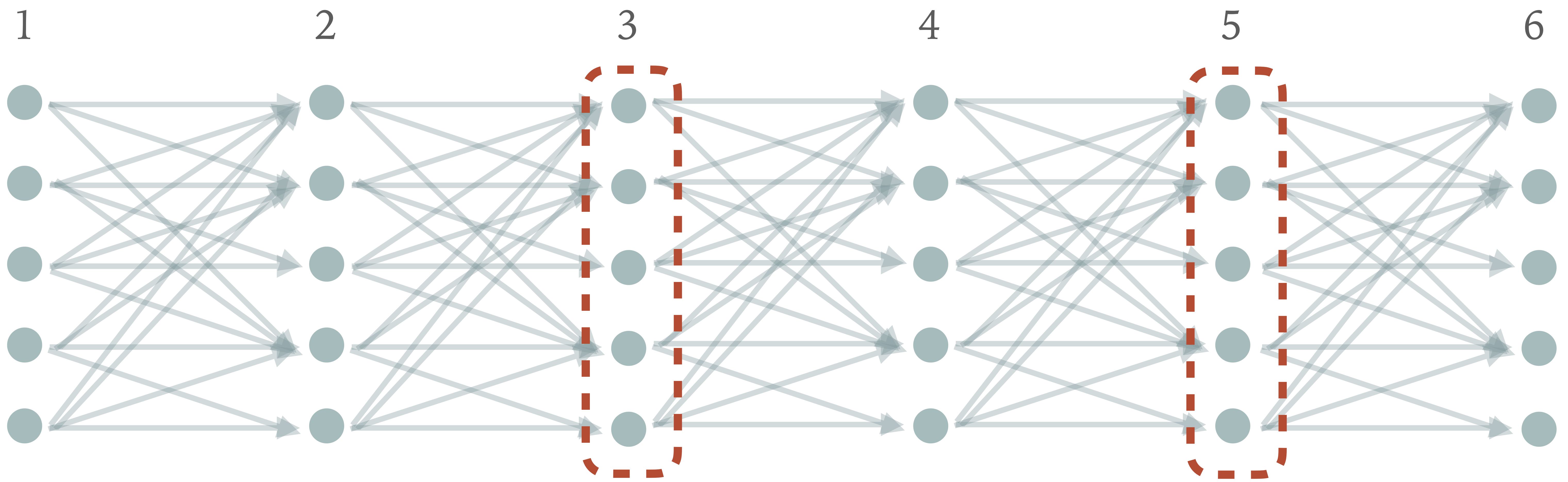
Optimization Problem

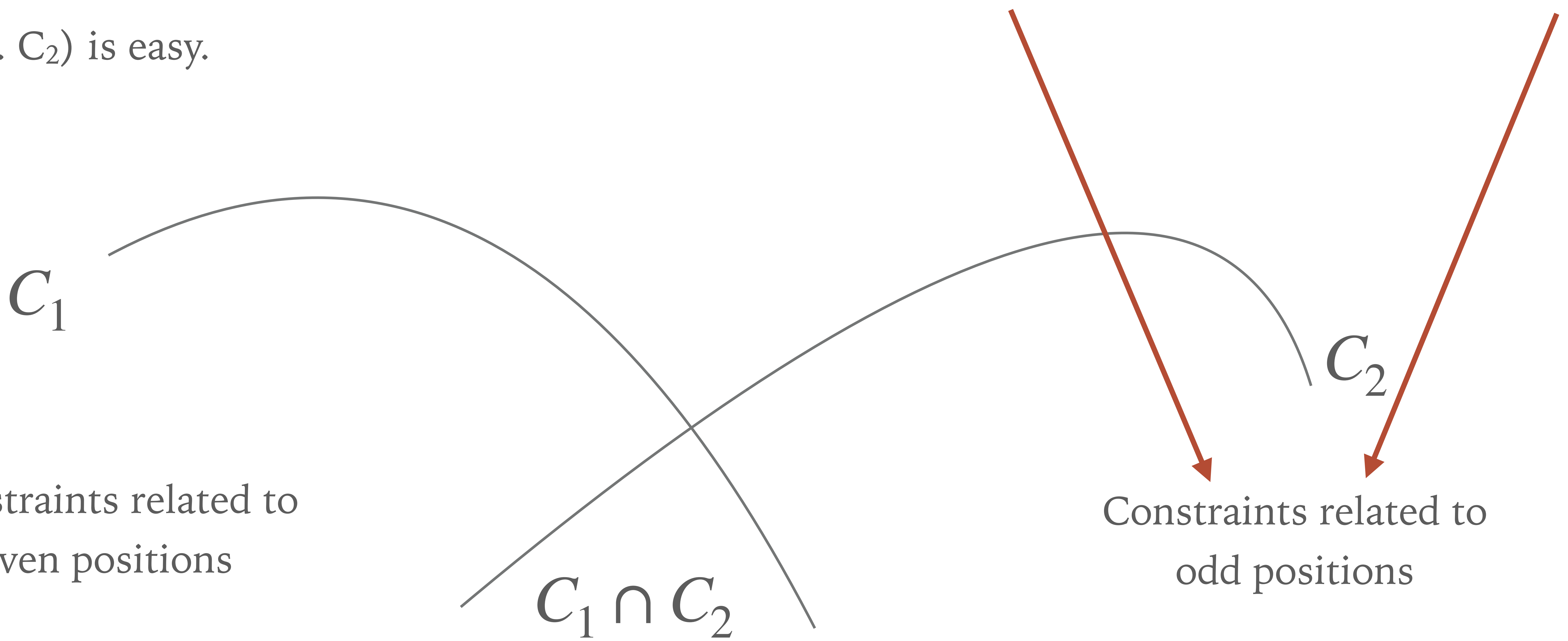
We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q}\in \mathbf{conv}\, Y} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q}\in C_1\cap C_2} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

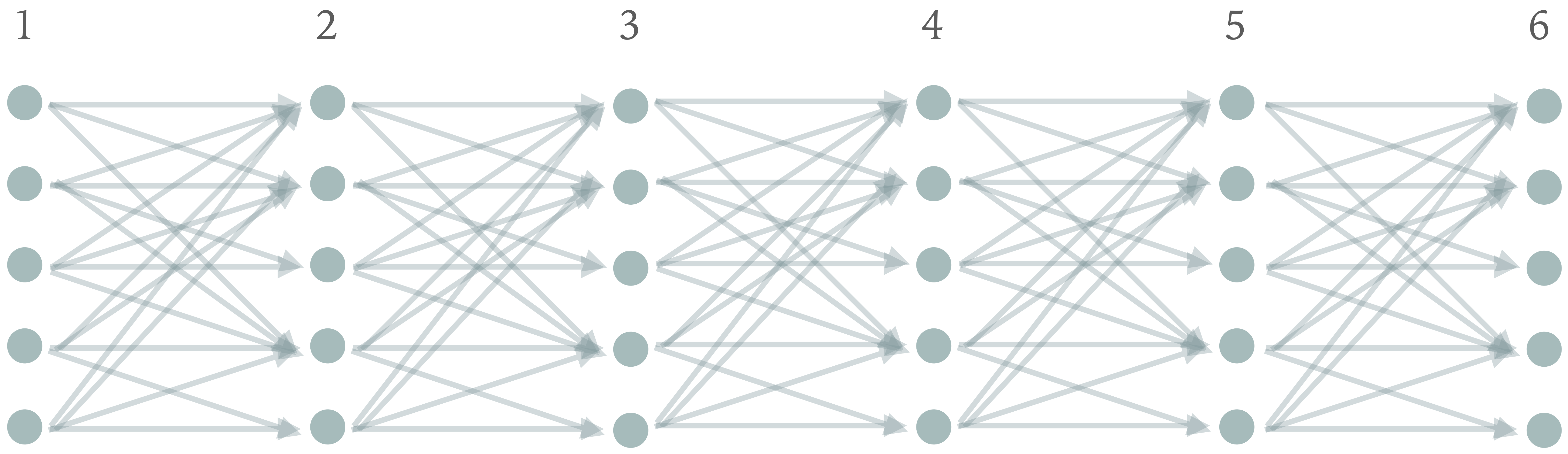
Optimization Problem

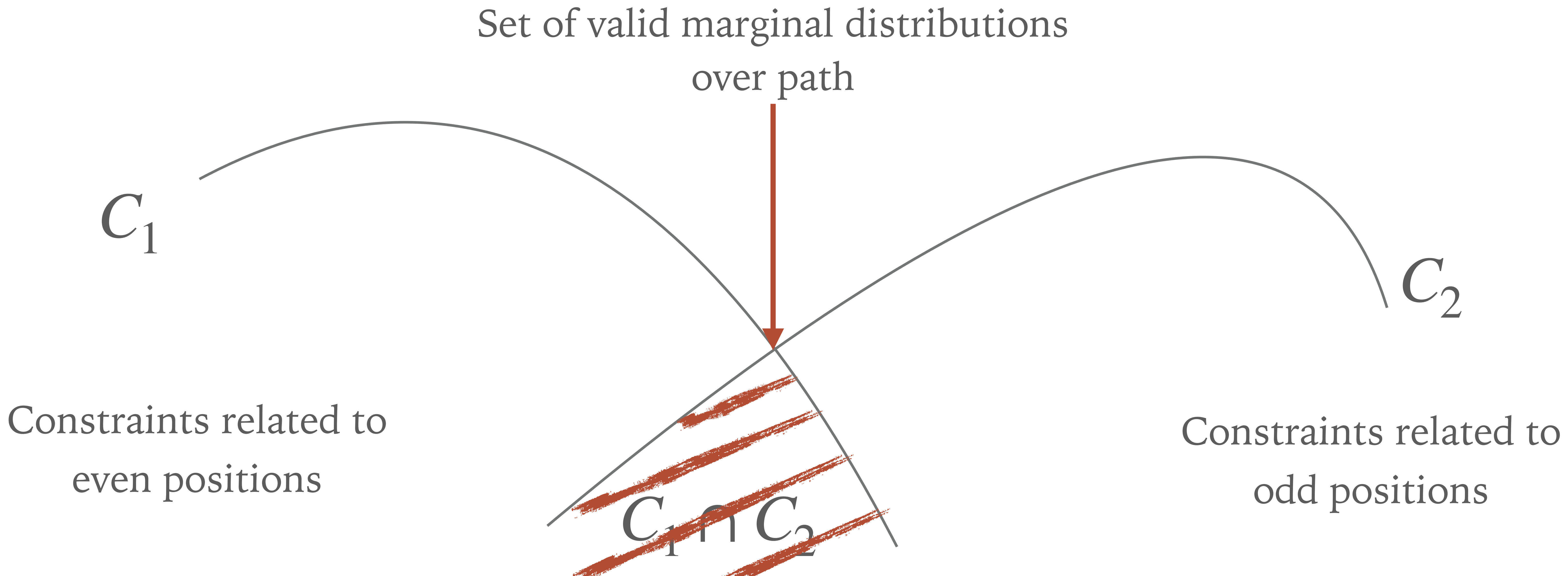
We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q}\in\mathbf{conv}\,Y} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q}\in C_1\cap C_2} \; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.




Set of valid marginal distributions over path

Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

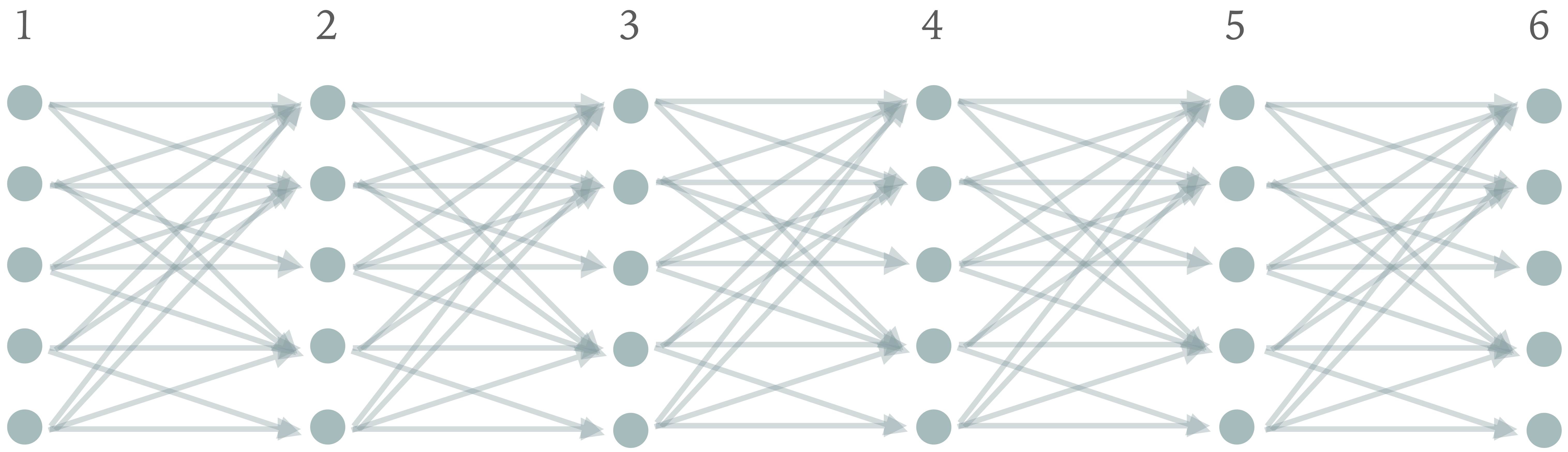
Optimization Problem

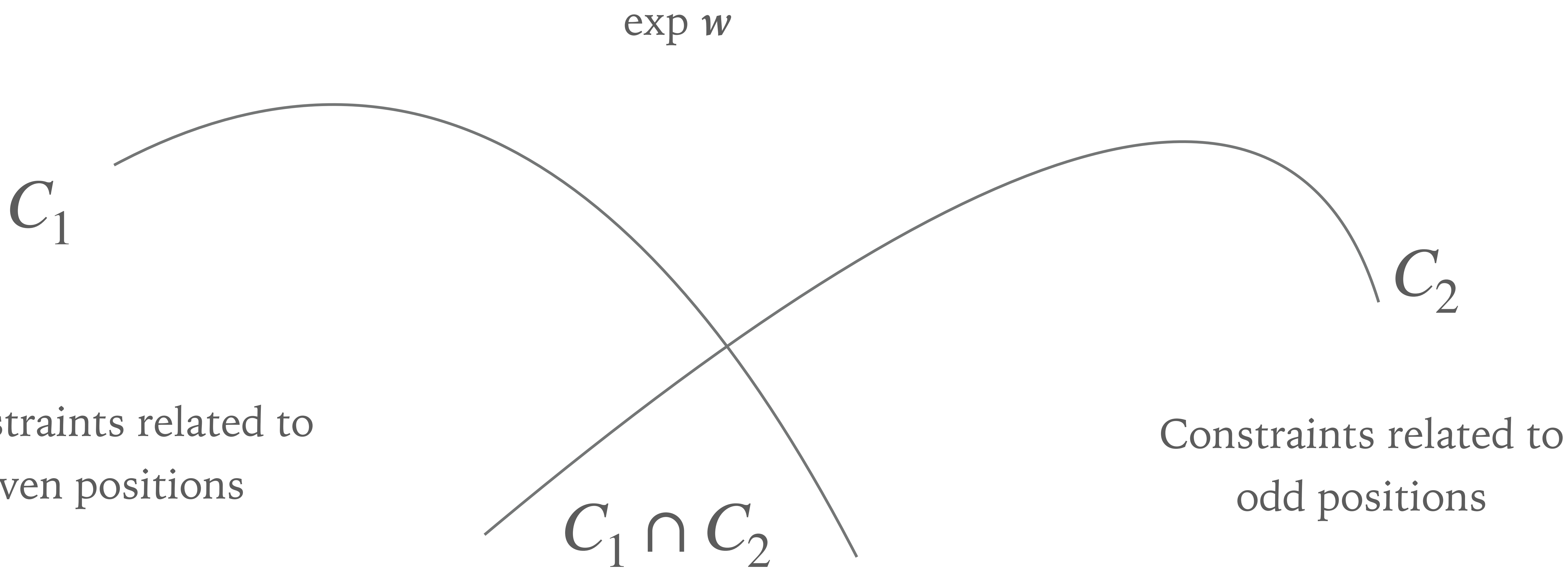
We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q}\in\mathbf{conv}\,Y} D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q}\in C_1\cap C_2} D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

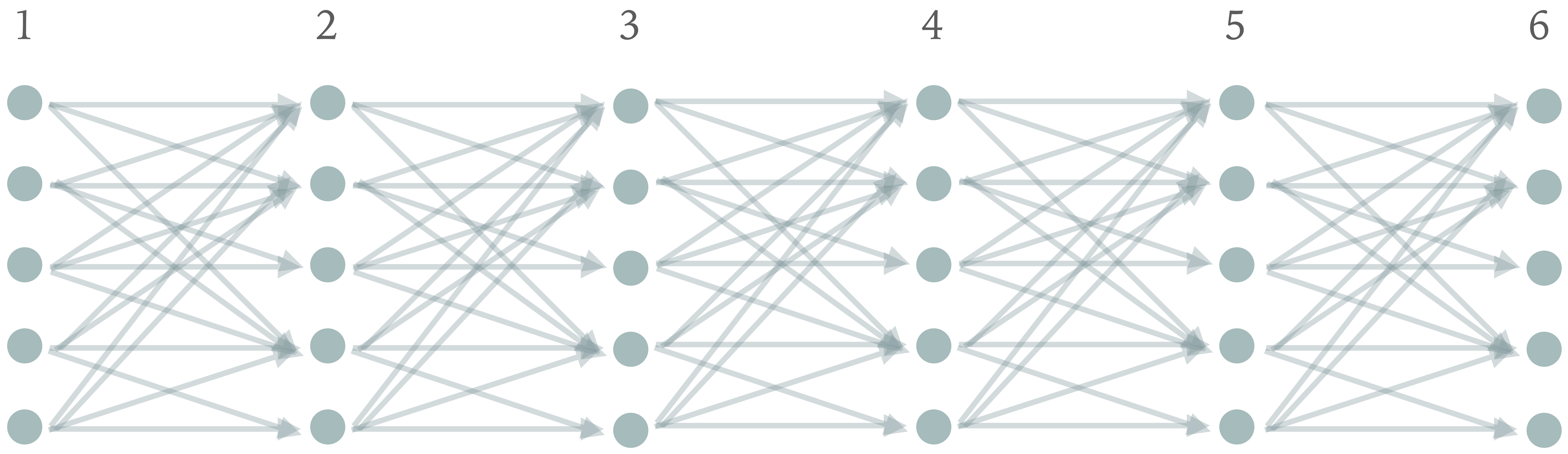
Optimization Problem

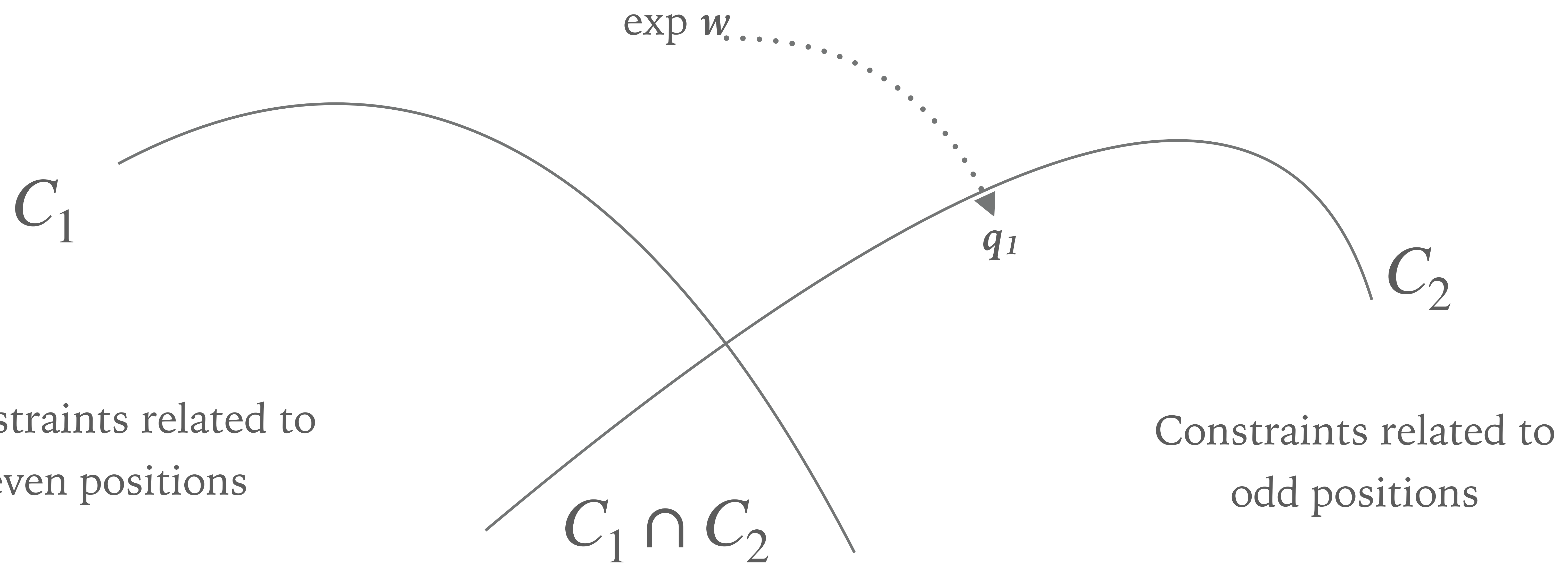
We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\mathbf{conv}\,Y}{\operatorname{argmin}}\; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

$$= \underset{\mathbf{q}\in C_1\cap C_2}{\operatorname{argmin}}\; D_{KL}(\mathbf{q} \,\|\, \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

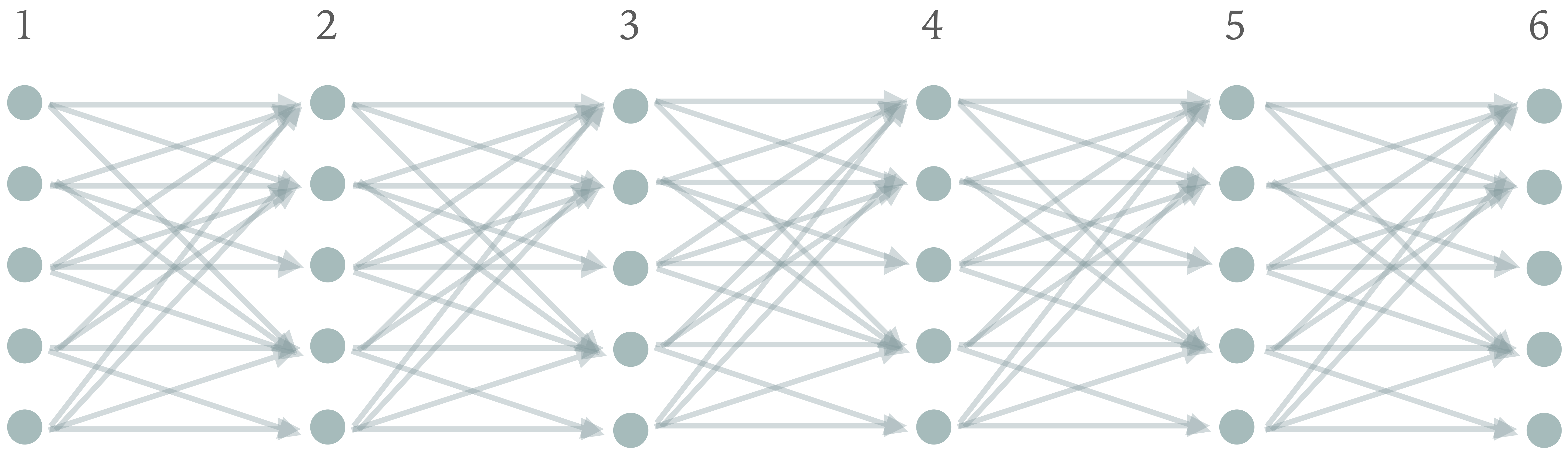
Optimization Problem

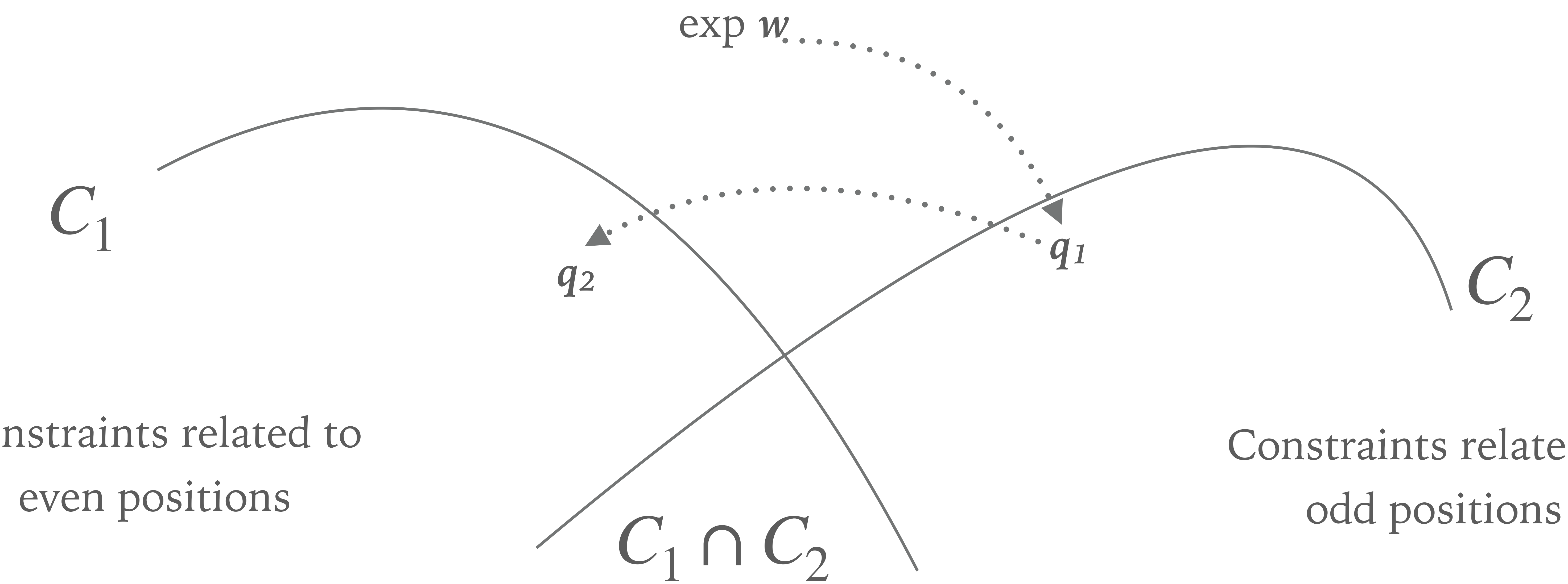
We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\textbf{conv}\,Y}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\,\|\,\exp\mathbf{w})$$

$$= \underset{\mathbf{q}\in C_1\cap C_2}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\,\|\,\exp\mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

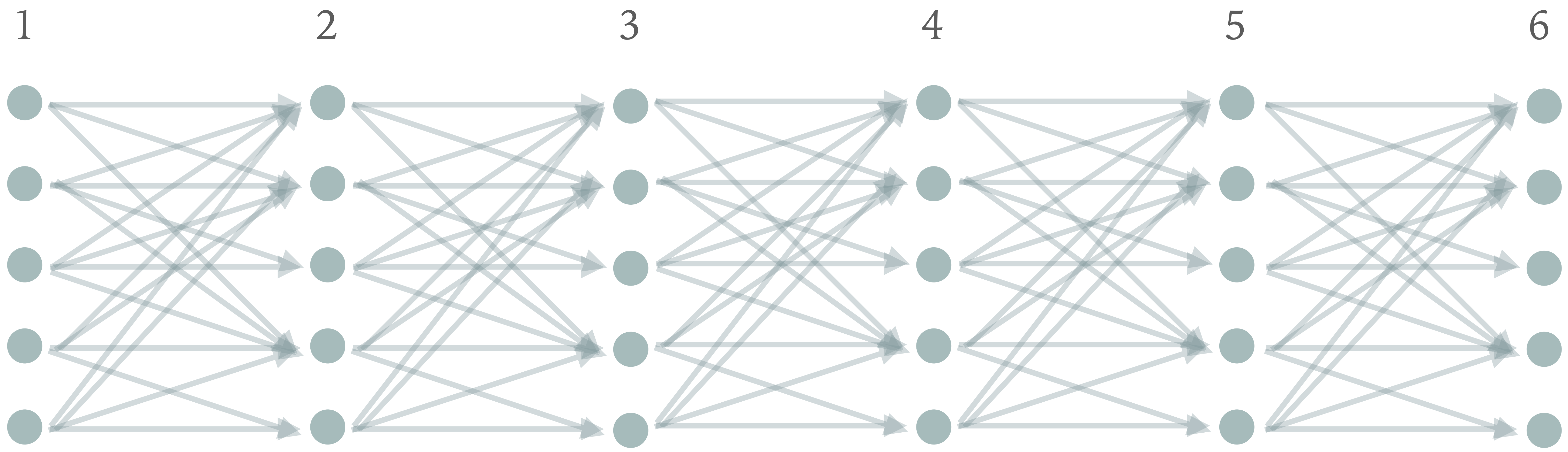
Optimization Problem

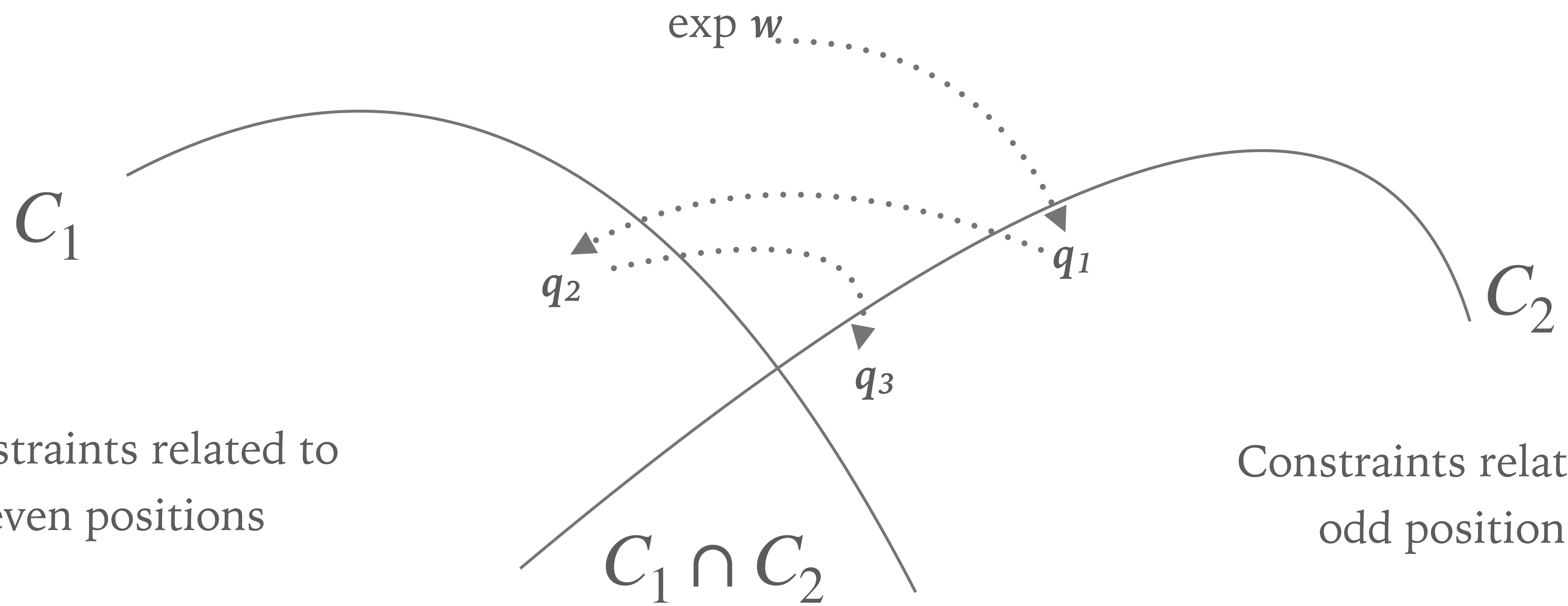
We decompose the KL projection into the projection into an intersection of sets:

$$\underset{\mathbf{q}\in\mathbf{conv}\,Y}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\parallel\exp\mathbf{w})$$

$$=\underset{\mathbf{q}\in C_1\cap C_2}{\operatorname{argmin}}\ D_{KL}(\mathbf{q}\parallel\exp\mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

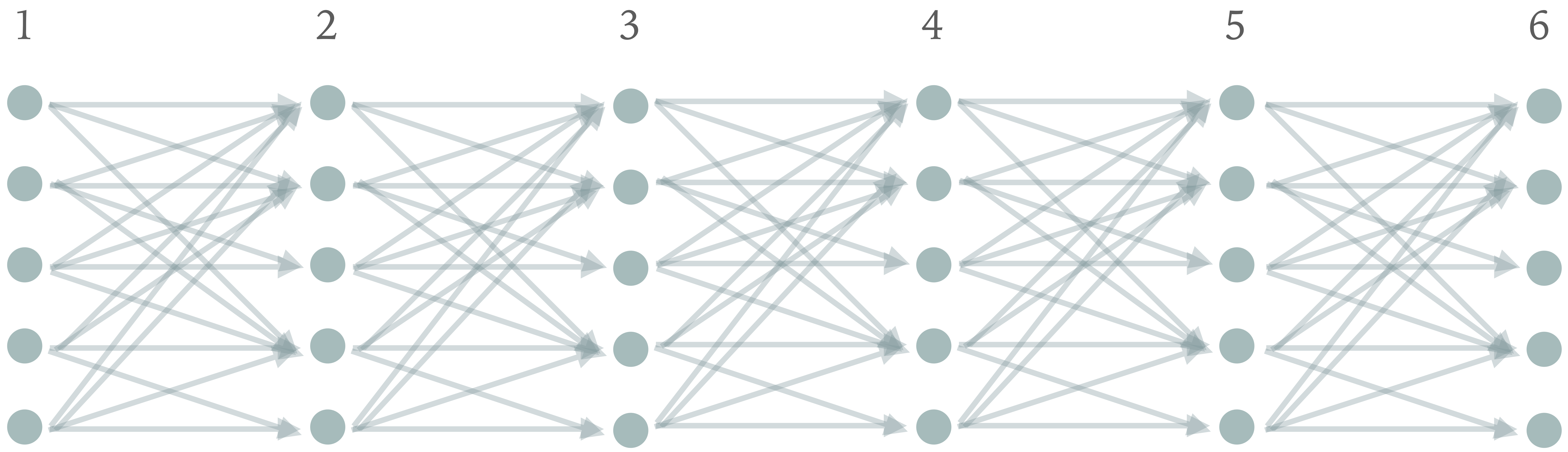
Optimization Problem

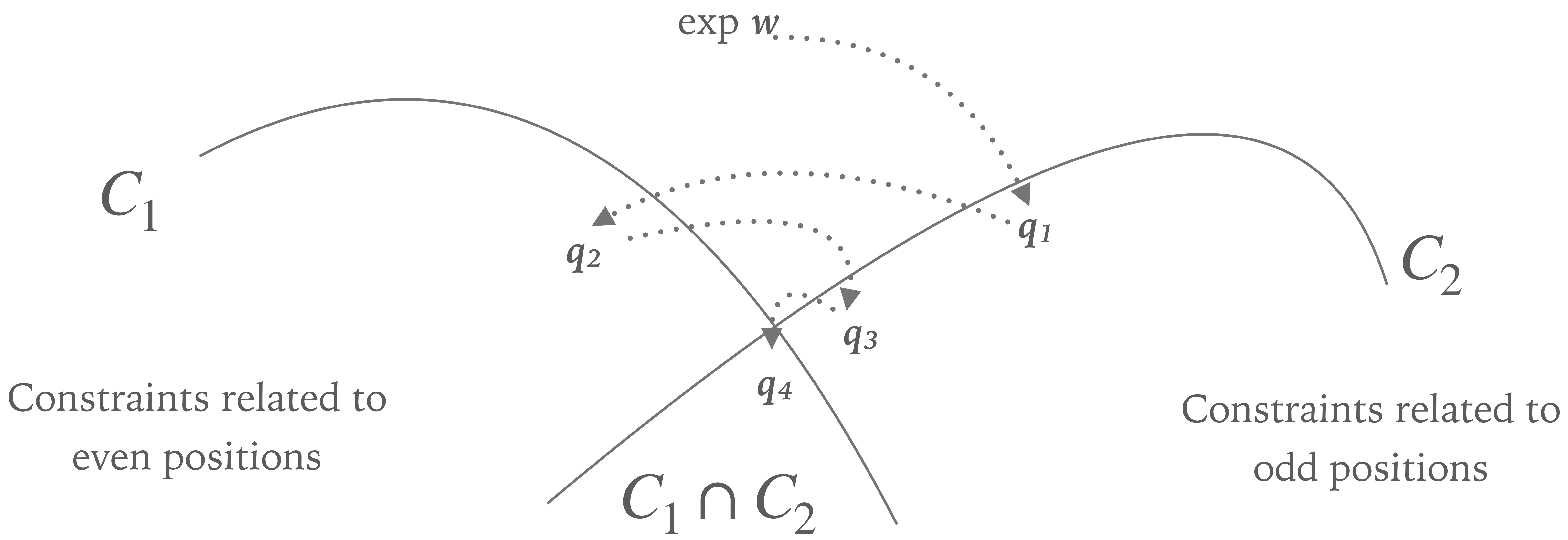
We decompose the KL projection into the projection into an intersection of sets:

$$\operatorname*{argmin}_{\mathbf{q} \in \mathbf{conv}\, Y} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

$$= \operatorname*{argmin}_{\mathbf{q} \in C_1 \cap C_2} D_{KL}(\mathbf{q} \parallel \exp \mathbf{w})$$

s.t. projection on C_1 (resp. C_2) is easy.





Constraints related to even positions

Constraints related to odd positions

BREGMAN CONDITIONAL RANDOM FIELDS

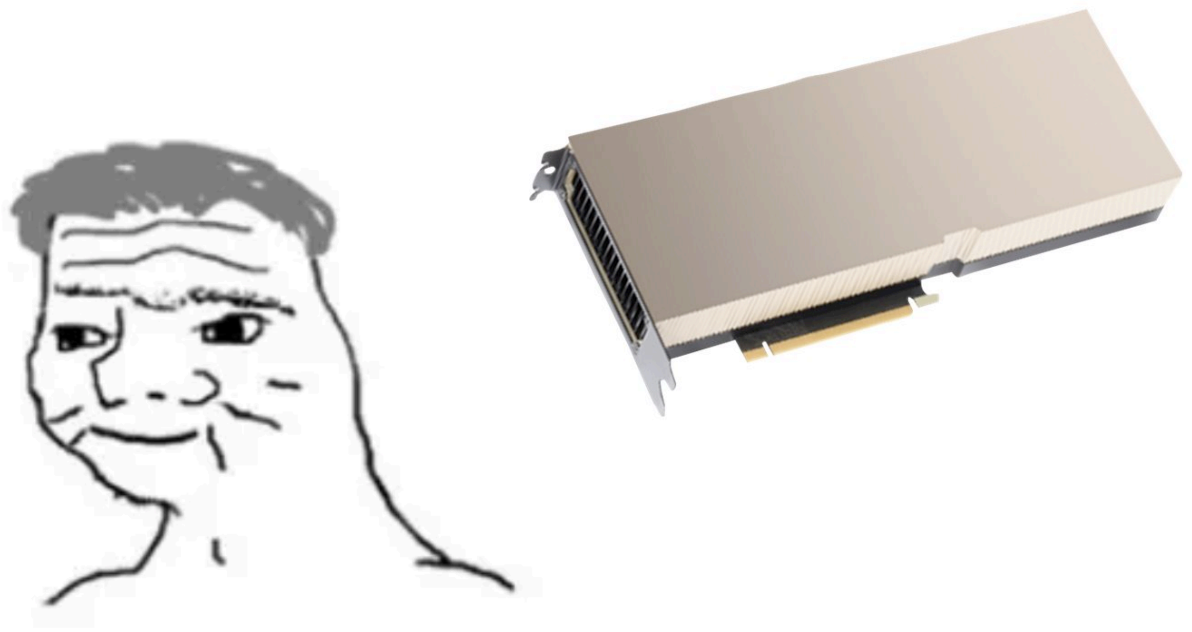
Caio Corro, Mathieu Lacroix, Joseph Le Roux

Main takeway

GPUs allows to **rethink well-known algorithms** to propose better parallelizable alternatives

TL;DR

- Novel distribution over sequence labelings using **mean regularization**
- Novel inference algorithm based on **iterative Bregman projections**
- **Supervised and weakly-supervised learning** using Fenchel-Young losses
- Many experimental results in the paper



GPUs GO BRRRRRRR

Experimental Results

- Faster on GPU than standard CRF for training and prediction ☺
- Somewhat slower than mean field for decoding ☹ but comparable speed for training ☺
- Better results than mean field when there are hard structural constraints (i.e. forbidden transitions) ☺
- Weakly-supervised learning scenario (not possible with mean field) ☺