Focused Belief Propagation for Query-Specific Inference

Anton Chechetka

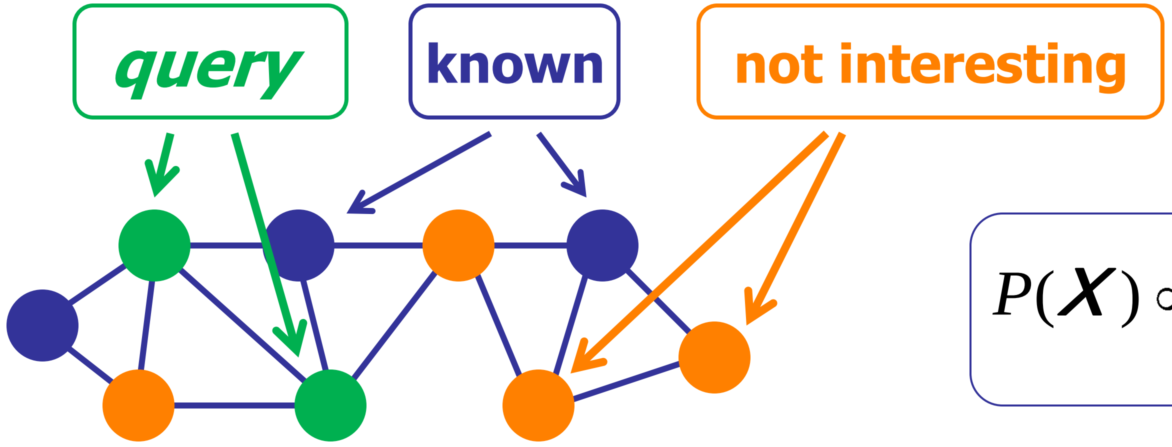
Carlos Guestrin

14 May 2010

Select Lab

Carnegie Mellon

Query-Specific inference problem



$$P(\boldsymbol{X}) \propto \prod_{ij \in E} f_{ij}(X_i X_j)$$

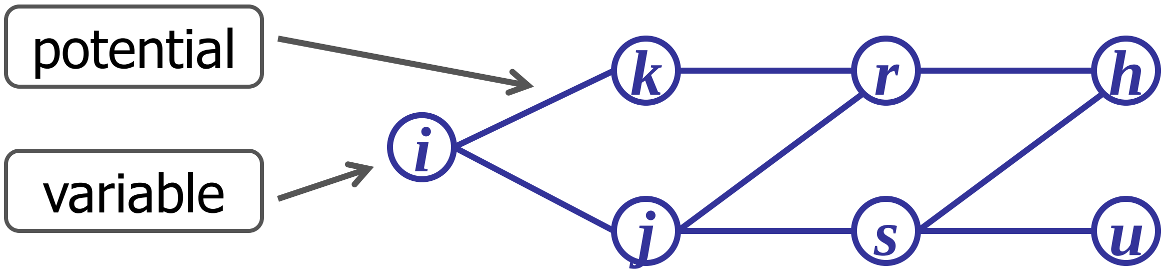
Using information about **the query**
to speed up convergence of **belief propagation**
for the query marginals

Graphical models

- ***Talk*** focus: ***pairwise Markov random fields***
- ***Paper: arbitrary factor graphs*** (more general)
- Probability is a product of ***local*** potentials

$$P(\boldsymbol{X}) \propto \prod_{ij \in E} f_{ij}(X_i X_j)$$

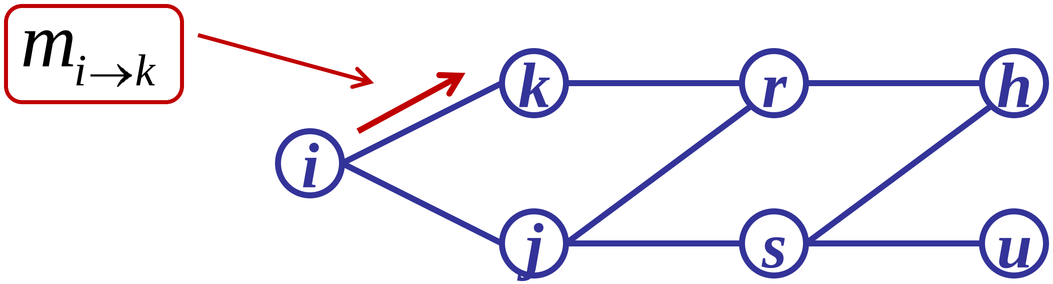
- Graphical structure



- ☺ Compact representation
- ☹ Intractable inference
  - ☺ Approximate inference often works well in practice

(loopy) Belief Propagation

- Passing messages along edges



- Variable belief:

$$\tilde{P}^{(t)}(x_i) \propto \prod_{ij\in E} m^{(t)}_{j\to i}(x_i)$$

- Update rule:

$$m^{(t+1)}_{j\to i}(x_i) = \sum_{x_j} f_{ij}(x_i x_j) \prod_{kj\in E, k\neq i} m^{(t)}_{k\to j}(x_j)$$

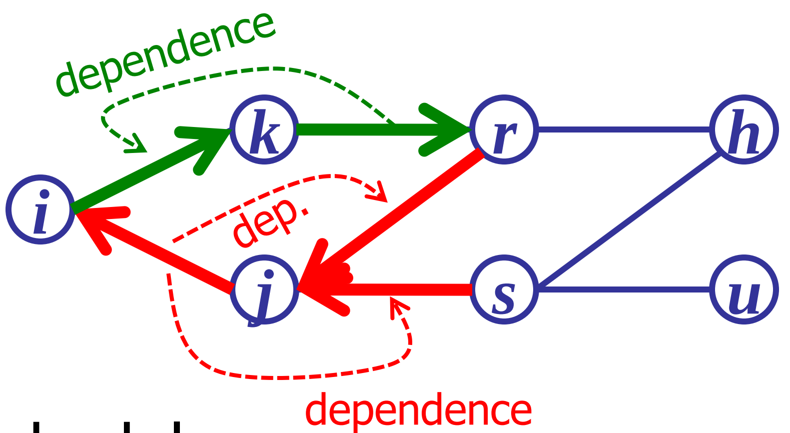
- Result: all single-variable beliefs

(loopy) Belief Propagation

- Update rule:

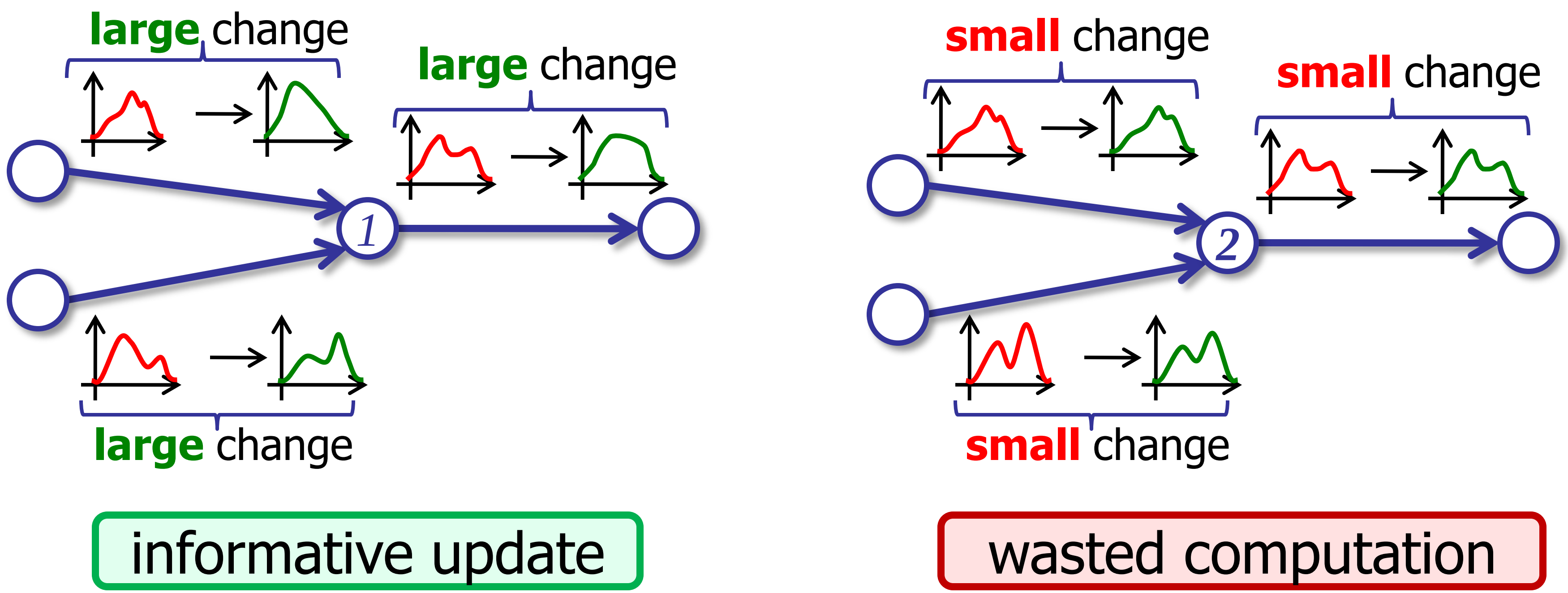
$$m_{j\to i}^{(t+1)}(x_i) = \sum_{x_j} f_{ij}(x_i x_j) \prod_{kj\in E, k\neq i} m_{k\to j}^{(t)}(x_j)$$

- Message dependencies are local:



- Round–robin schedule
  - Fix message order
  - Apply updates in that order until convergence

Dynamic update prioritization



- ***Fixed*** update ***sequence*** is not the best option
- Dynamic ***update scheduling*** can speed up convergence
  - Tree-Reweighted BP [Wainwright et. al., AISTATS 2003]
  - Residual BP [Elidan et. al. UAI 2006]
- Residual BP ➔ apply the ***largest change*** first

Residual BP [Elidan et. al., UAI 2006]

- Update rule:

$$\text{new} \leftarrow m_{j\to i}^{(NEW)}(x_i) = \sum_{x_j} f_{ij}(x_i x_j) \prod_{kj\in E, k\neq i} m_{k\to j}^{(OLD)}(x_j) \rightarrow \text{old}$$

- Pick edge with ***largest residual***

$$\max \left\| m_{j\to i}^{(NEW)} - m_{j\to i}^{(OLD)} \right\|$$

- Update $m_{j\to i}^{(OLD)} \leftarrow m_{j\to i}^{(NEW)}$

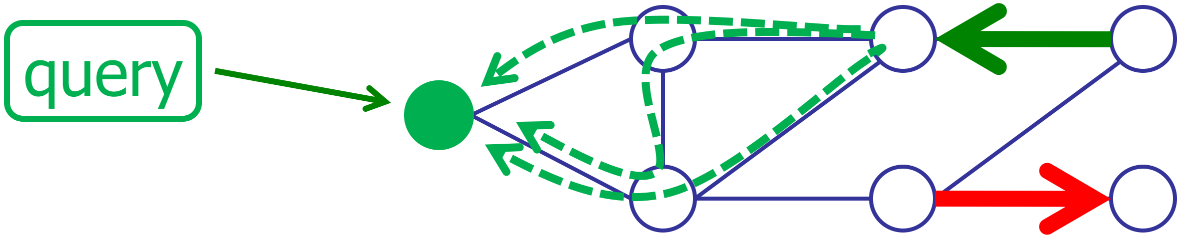
More effort on the difficult parts of the model ☺

But no query ☹

Our contributions

- Using ***weighted*** residuals to prioritize updates
- Define message weights reflecting the ***importance*** of the message ***to the query***
- Computing importance weights efficiently
- Experiments: faster convergence on large relational models

Why edge importance weights?



residual < **residual**
which to update??

- Residual BP updates →
- ***no influence*** on the query
- wasted computation

- want to update ←
- ***influence*** on the query ***in the future***

Our work → max approx. ***eventual effect on P(query)***

Residual BP → max ***immediate residual reduction***

Query-Specific BP

- Update rule:

new ← $$m^{(NEW)}_{j\to i}(x_i) = \sum_{x_j} f_{ij}(x_i x_j) \prod_{kj\in E, k\neq i} m^{(OLD)}_{k\to j}(x_j)$$ → old

- Pick edge with $\max \left\| m^{(NEW)}_{j\to i} - m^{(OLD)}_{j\to i} \right\|$ → edge importance

- Update $m^{(OLD)}_{j\to i} \leftarrow m^{(NEW)}_{j\to i}$

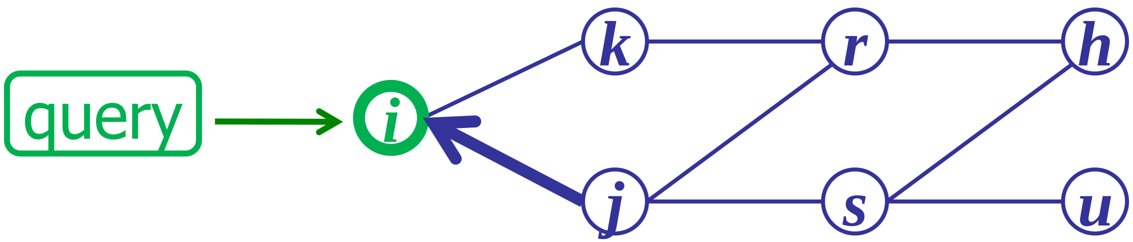
the only change!

Rest of the talk:
defining and computing edge importance

Edge importance base case

- Pick edge with $\max \left\| m_{j\to i}^{(NEW)} - m_{j\to i}^{(OLD)} \right\| \times A_{j\to i}$

approximate eventual update effect on P(**Q**)



Base case: edge ***directly connected*** to the query

$\mathbf{A_{j\to i}=??}$

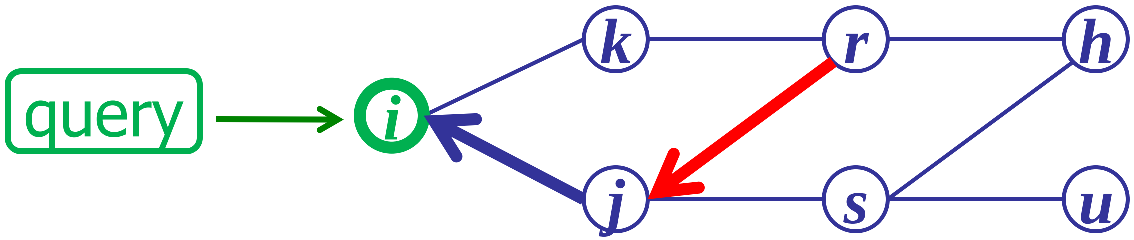
change in query belief — change in message

$$||P^{(NEW)}(\mathbf{Q}) - P^{(OLD)}(\mathbf{Q})|| \le ||\mathbf{m}_{j\to i}^{(NEW)} - \mathbf{m}_{j\to i}^{(OLD)}|| \times \mathbf{1}$$

$||\Delta P(\mathbf{Q})||$ — tight bound — $||\Delta \mathbf{m}_{j\to i}||$

sense learn act

Edge importance one step away



Edge ***one step away*** from the query: $\mathbf{A_{r\to j}=??}$

$$||\Delta P(\mathbf{Q})|| \le ||\Delta \mathbf{m}_{j\to i}||$$

$$\le ||\Delta \mathbf{m}_{r\to j}|| \times \sup \left\| \frac{\partial \mathbf{m}_{j\to i}}{\partial \mathbf{m}_{r\to j}} \right\|$$

change in query belief

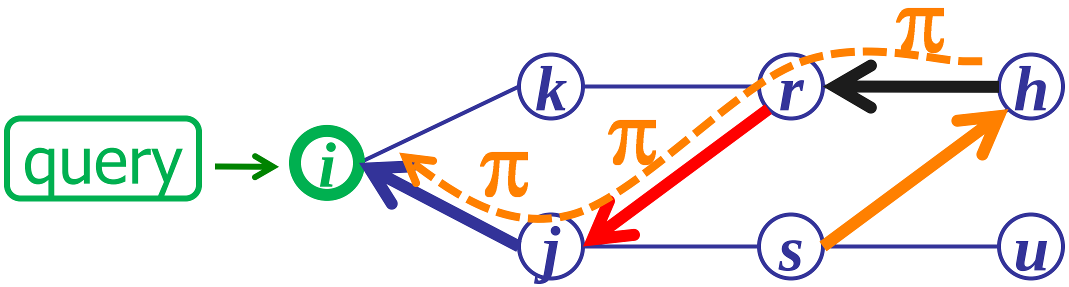
change in message

message importance

over values of **all other messages**

can compute **in closed form** looking at only f_{ji} [Mooij, Kappen; 2007]

Edge importance general case



Base case: $A_{j\to i}=1$

One step away:

$$A_{r\to j} = \sup \left\| \frac{\partial m_{j\to i}}{\partial m_{r\to j}} \right\|$$

$$\|\Delta P(Q)\| \le \|\Delta m_{s\to h}\| \times \sup \left\| \frac{\partial P(Q)}{\partial m_{s\to h}} \right\|$$

Generalization?

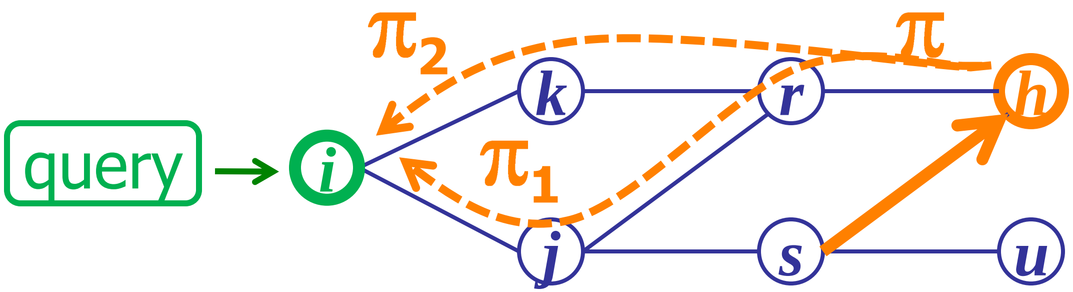
☹ ***expensive*** to compute

☹ bound may be ***infinite***

$$\sup \left\| \frac{\partial P(Q)}{\partial m_{s\to h}} \Big|_{\pi} \right\| \le \underbrace{\sup \left\| \frac{\partial m_{h\to r}}{\partial m_{s\to h}} \right\| \times \sup \left\| \frac{\partial m_{r\to j}}{\partial m_{h\to r}} \right\| \times \sup \left\| \frac{\partial m_{j\to i}}{\partial m_{r\to j}} \right\|}$$

sensitivity(π): max impact *along the path* π

Edge importance general case



$$\sup \left\| \frac{\partial P(Q)}{\partial m_{s\to h}} \bigg|_{\pi} \right\| \leq \underbrace{\sup \left\| \frac{\partial m_{h\to r}}{\partial m_{s\to h}} \right\| \times \sup \left\| \frac{\partial m_{r\to j}}{\partial m_{h\to r}} \right\| \times \sup \left\| \frac{\partial m_{j\to i}}{\partial m_{r\to j}} \right\|}$$

sensitivity(π): max impact *along the path* π

$$A_{s\to h} = \max_{\text{all paths } \pi \text{ from } h \text{ to query}} \text{sensitivity}(\pi)$$

There are **a lot** of paths in a graph,
trying out every one is intractable ☹

Efficient edge importance computation

A = max **all paths π from ← to query** **sensitivity(π)**

There are **a lot** of paths in a graph,
trying out every one is intractable ☹

always decreases as the path grows

Dijkstra's (shortest paths) **alg.**
will ***efficiently*** find max-sensitivity paths
for every edge ☺

sensitivity(**h→r→j→i**) =

decomposes into
individual edge contributions

Query-Specific BP

- Run Dijkstra's alg **starting at query** to get edge weights

$$A_{j\to i} = \max_{\text{all paths } \pi \text{ from } i \text{ to query}} \text{sensitivity}(\pi)$$

More effort on the difficult **and relevant** parts of the model

$$\max \| m_{j\to i} - m_{j\to i} \| \times A_{j\to i}$$

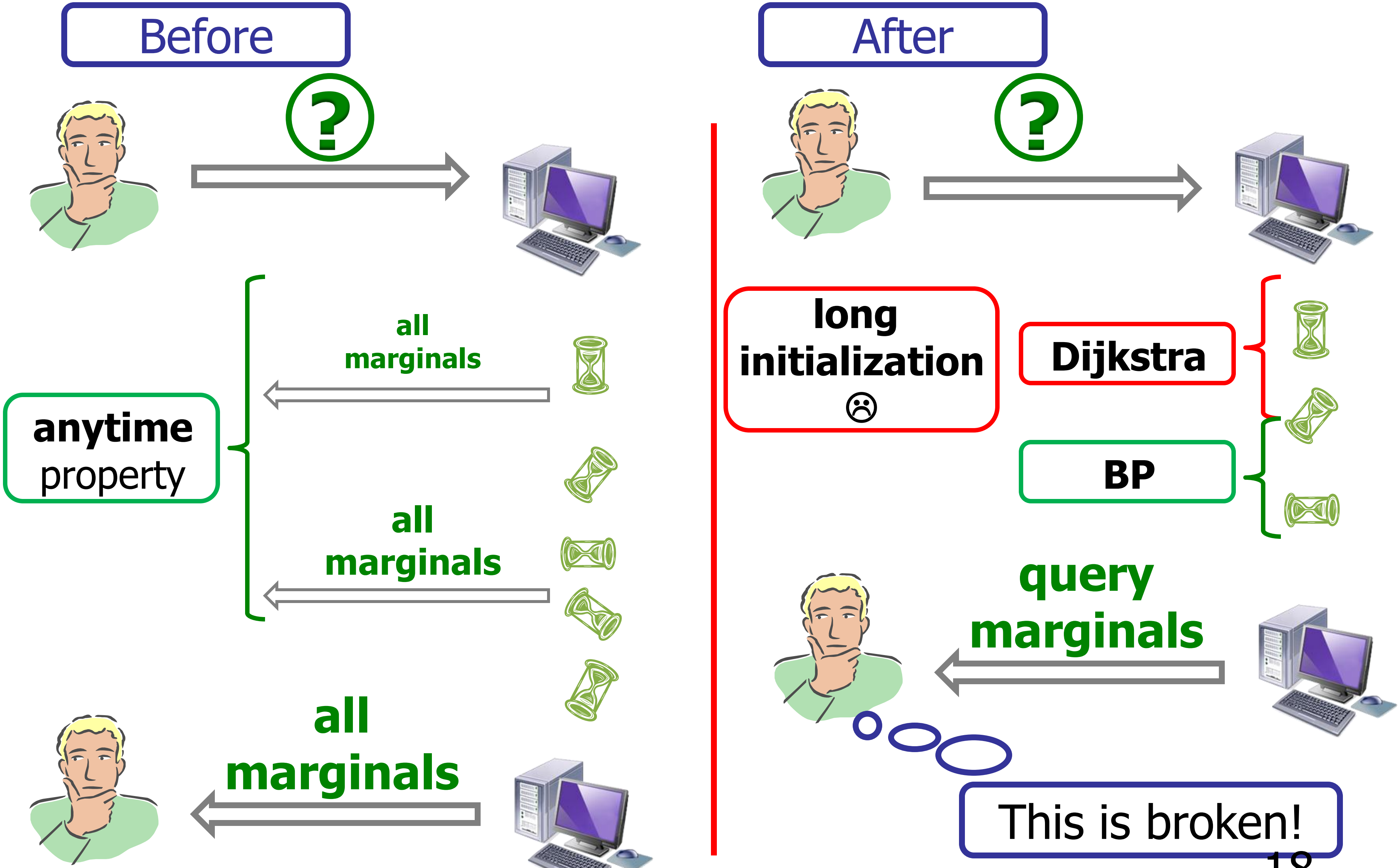
- U

Takes into account not only ***graphical structure***, but also ***strength of dependencies***

Outline

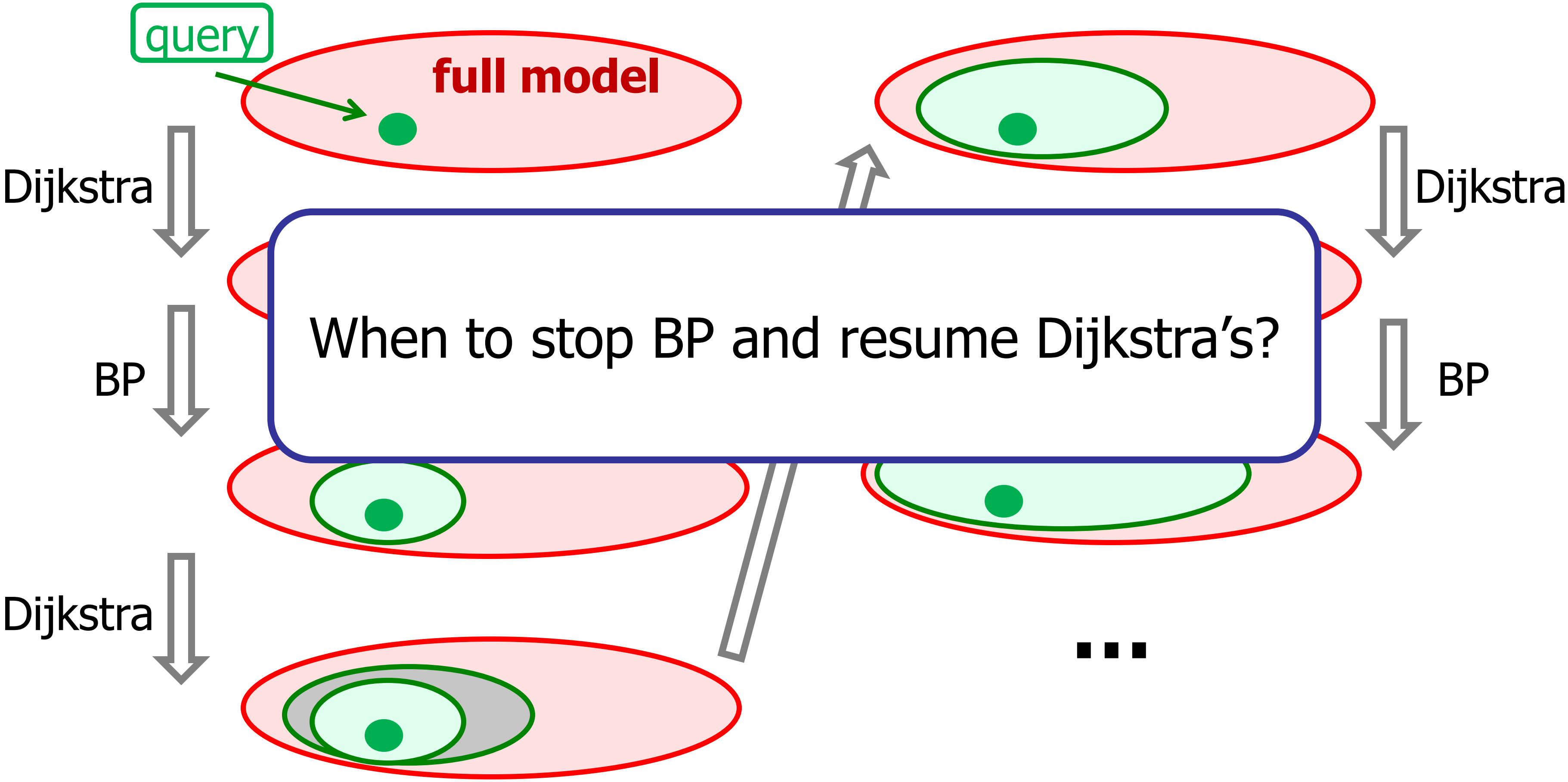
- Using ***weighted*** residuals to prioritize updates
- Define message weights reflecting the ***importance*** of the message ***to the query***
- Computing importance weights efficiently
  - As an initialization step before residual BP
- Restore ***anytime behavior*** of residual BP
- Experiments: faster convergence on large relational models

Big picture


Before
After
?
?
all
marginals
anytime
property
all
marginals
all
marginals
long
initialization
☹
Dijkstra
BP
query
marginals
This is broken!

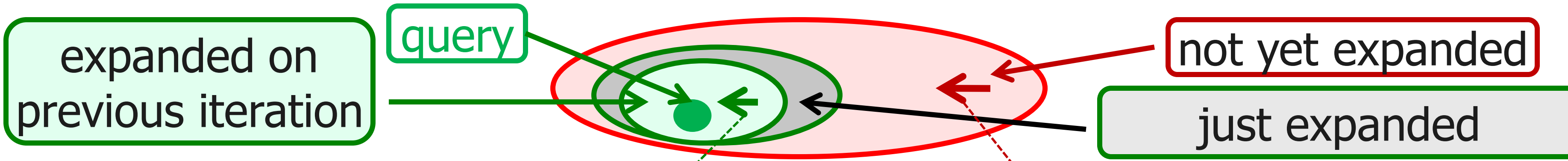
Interleaving BP and Dijkstra's

- Dijkstra's **expands** the **highest weight** edges **first**
  - Can pause it at any time and get the **most relevant submodel**



Interleaving

- Dijkstra's expands the **highest weight** edges **first**



min$_{\leftarrow \in}$ **expanded edges** **A**$_{\leftarrow}$ $\geq$ **A**$_{\leftarrow}$

suppose $M \geq \max_{j \to i \in ALL\ EDGES} \left\| m^{(NEW)}_{j \to i} - m^{(OLD)}_{j \to i} \right\|$

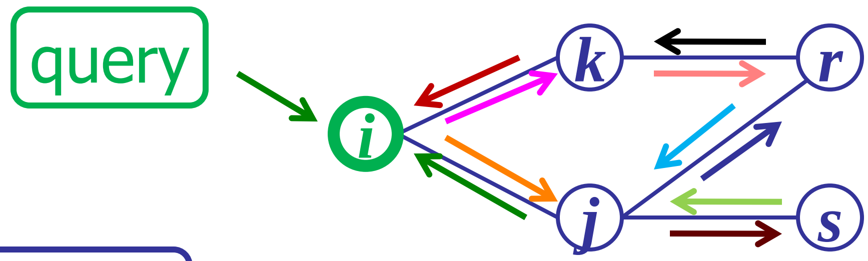
actual priority of ← upper bound on ← priority

$\max_{j \to i \in EXPANDED} \left\| m^{(NEW)}_{j \to i} - m^{(OLD)}_{j \to i} \right\| \times A_{j \to i}$ **M** × **min**$_{\leftarrow \in}$ **expanded** **A**$_{\leftarrow}$

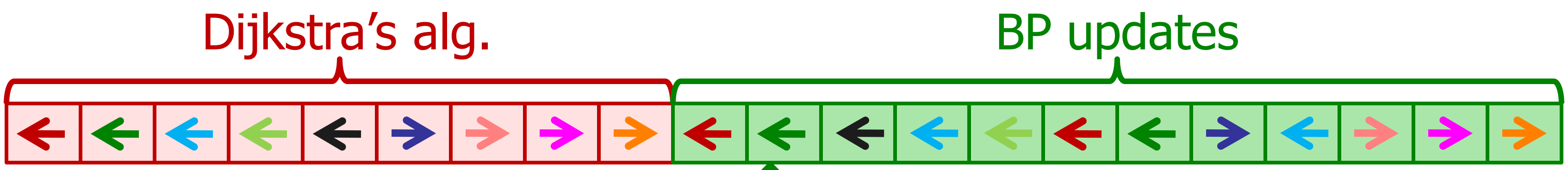
⇩

no need to expand further at this point

Any-Time query-specific BP

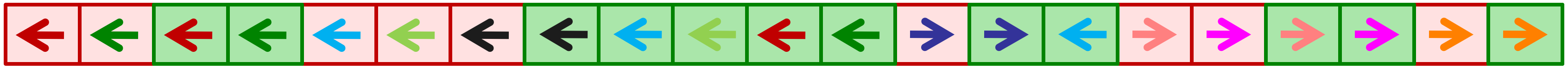


Query-specific BP:



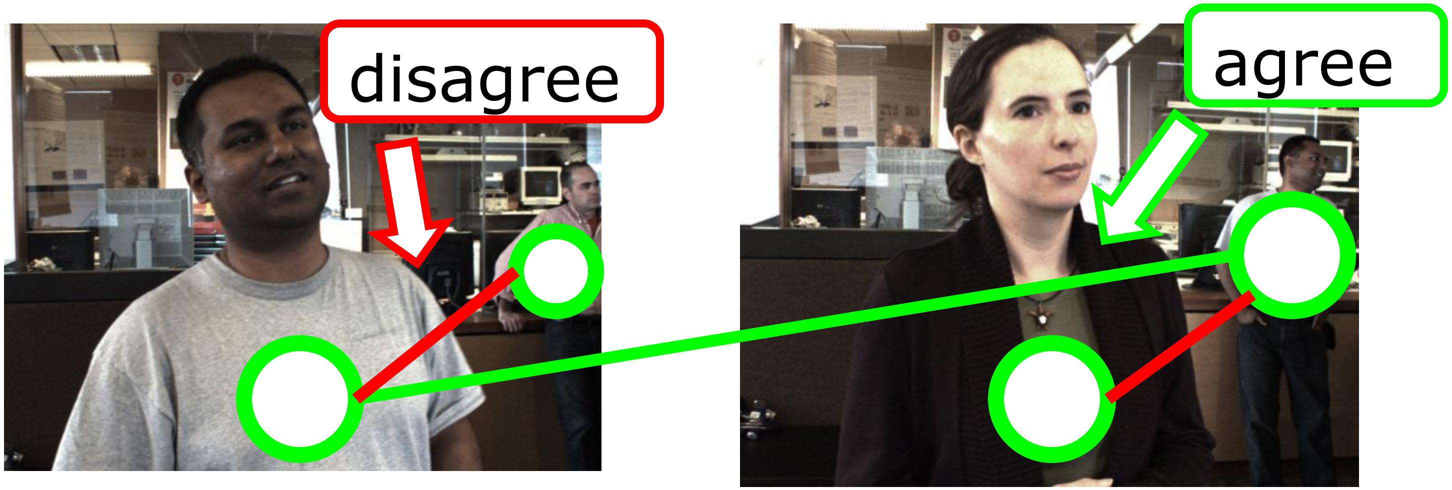
same BP update **sequence!**

Anytime QSBP:

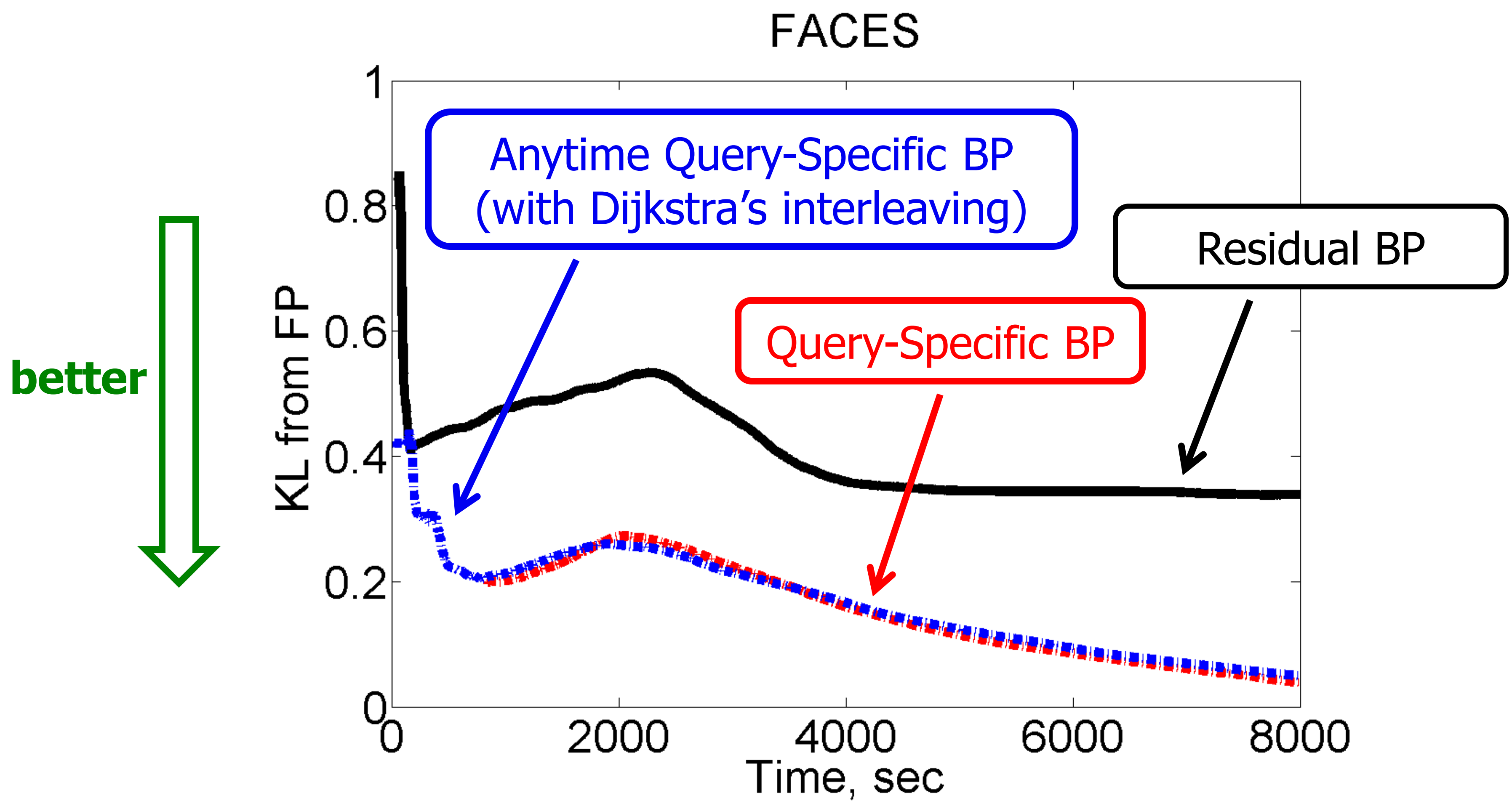


Experiments – the setup

- Relational model: semi-supervised people recognition in collection of images [with Denver Dash and Matthai Philipose @ Intel]
  - One variable per person
  - Agreement potentials for people who look alike
  - Disagreement potentials for people in the same image
  - 2K variables, 900K factors
- Error measure: KL from the fixed point of BP



Experiments – convergence speed



Conclusions

- Prioritize updates by ***weighted*** residuals
  - Takes ***query info*** into account
- Importance weights depend on both the ***graphical structure*** and ***strength of*** local ***dependencies***
- ***Efficient computation*** of importance weights
- Much ***faster convergence*** on large relational models

Thank you!