Northeastern

CS 7140: ADVANCED MACHINE LEARNING

Recap: Structured Prediction

- A prediction function $f : \mathscr{X} \longrightarrow \mathscr{Y}$, where the output $\mathscr{Y}$ domain is structured
  $y^\star = f(x) = \text{argmax}_y g(x, y)$ $\boxed{\text{MAP inference}}$

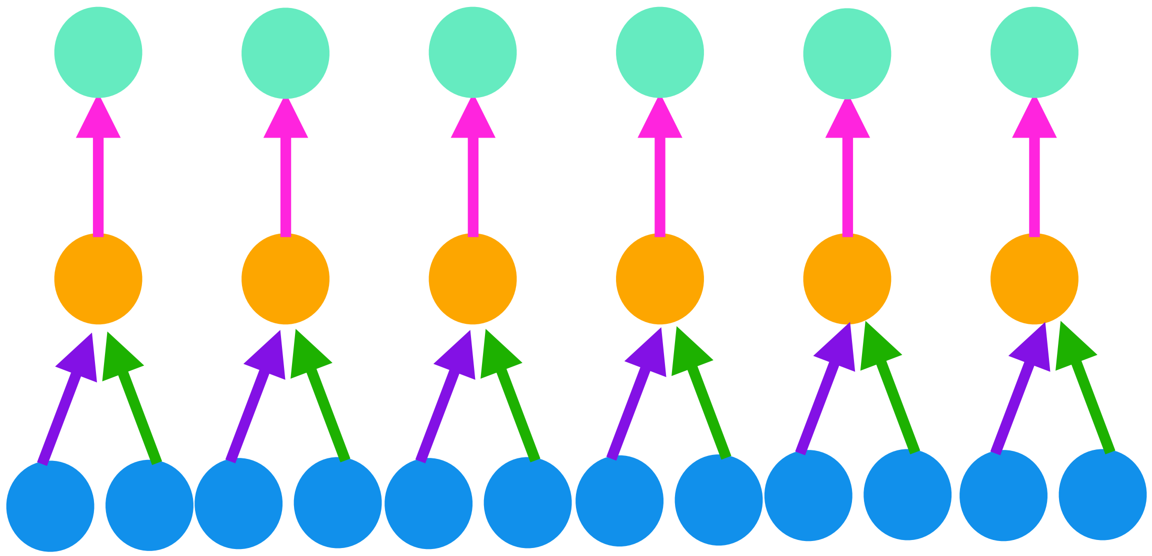
- $g(x, y) = p(y \mid x)$ when the model is probabilistic

- Graph Cut, Local Search, Branch and Bound, Lagrangian Regularization.

Recap: CNN/RNN

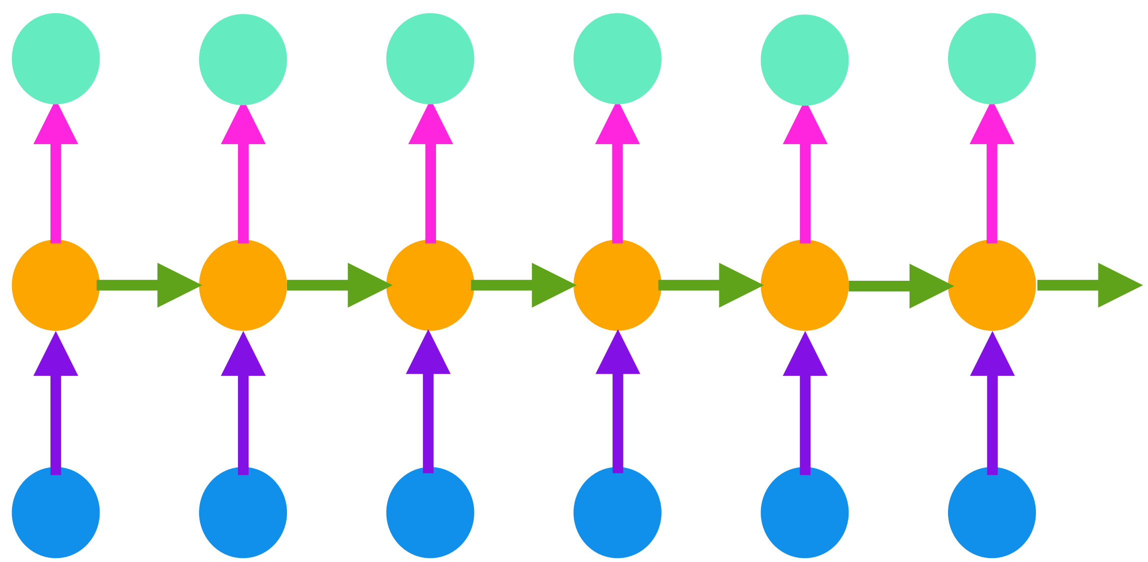
CNN

sharing weights across filters



RNN

sharing weights across transitions



INFERENCE AND HOPFIELD NETWORK

Exact Inference Problems

- The normalization problem:

$$P(\mathbf{x}) = \prod_{m=1}^{M} f(\mathbf{x}_m) \quad Z = \sum_{\mathbf{x}} \prod_{m=1}^{M} f(\mathbf{x}_m)$$

- The marginalization problem:

$$Z_n(x_n) = \sum_{\{x_{n'}\},\ n' \neq n} P(\mathbf{x})$$

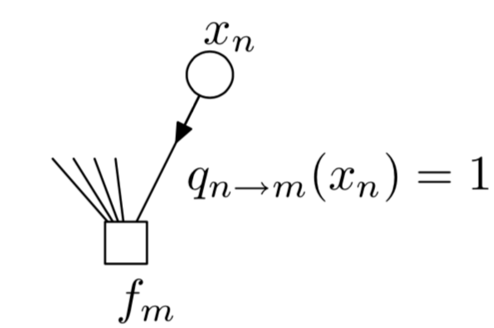
Message Passing

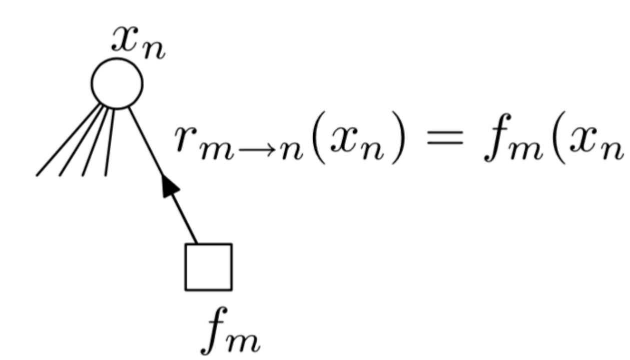
From variable to factor:

$$q_{n\to m}(x_n) = \prod_{m' \in \mathcal{M}(n)\backslash m} r_{m'\to n}(x_n). \tag{26.11}$$

From factor to variable:

$$r_{m\to n}(x_n) = \sum_{\mathbf{x}_m \backslash n} \left(f_m(\mathbf{x}_m) \prod_{n' \in \mathcal{N}(m)\backslash n} q_{n'\to m}(x_{n'}) \right). \tag{26.12}$$

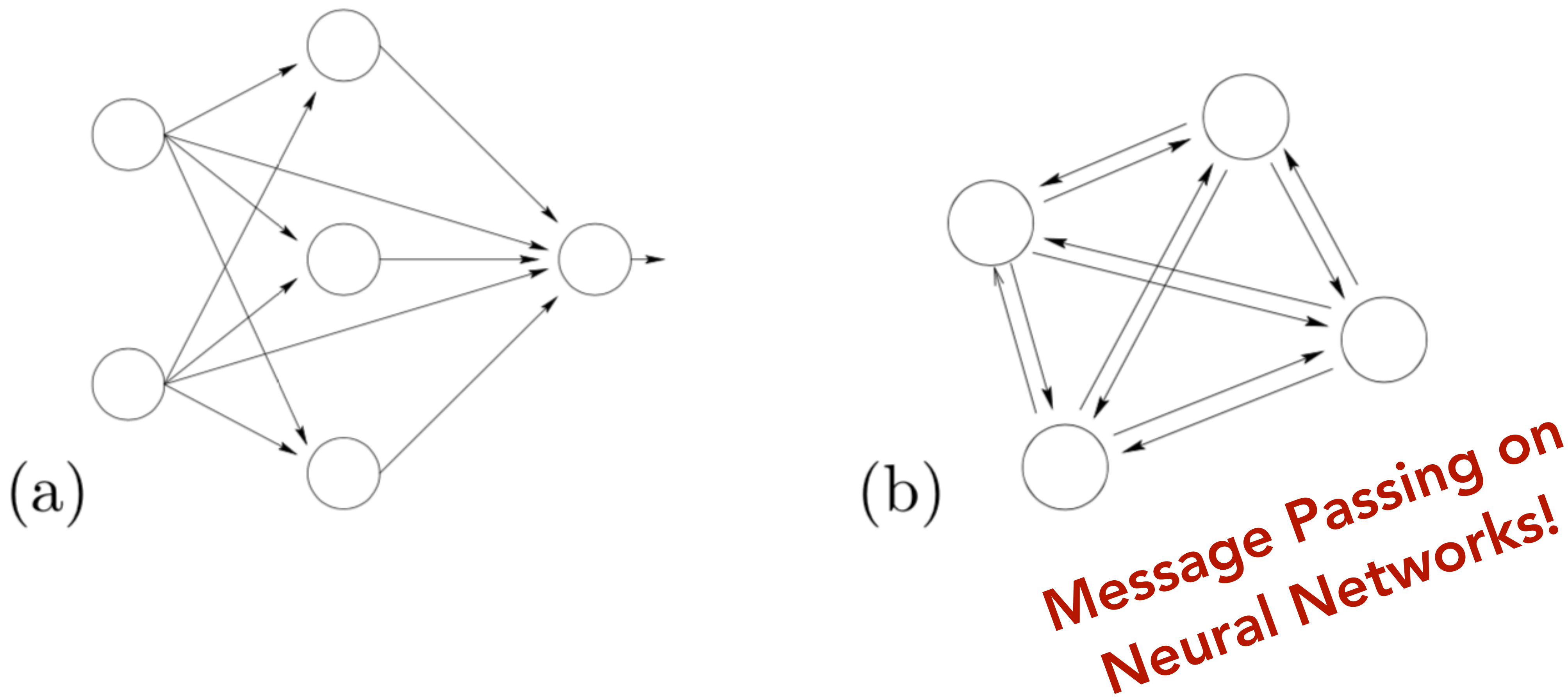




Two types of messages passing along the edges:

- messages $q_{n\to m}$ from variable nodes to factor nodes
- messages $r_{m\to n}$ n from factor nodes to variable nodes.

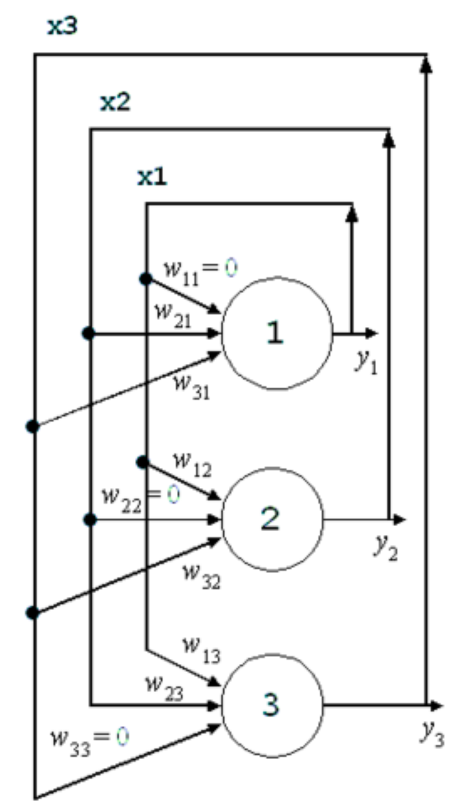
Feedforward/Feedback Networks



- **Feedforward**: all the connections are directed and form a directed acyclic graph
- **Feedback**: any network that is not a feedforward network
- **Applications**: Associated Memory, Optimization Problem

Binary Hopfield Network

- **Weights** w_{ij} denotes the connection from neuron i to neuron j
- **Architecture** symmetric, bidirectional connections $w_{ij} = w_{ji}$, no self-connections $w_{ii} = 0$
- **Activity rule** single neuron update
$$x(a) = \begin{cases} 1 & a \geq 0 \\ -1 & a < 0 \end{cases}$$
- **Updates** activations $a_i = \sum_j w_{ij} x_j$
- **Learning rule** make a set of desired memory $\{x^{(n)}\}$ be stable states, set weights $w_{ij} = \eta \sum_{ij} x_i^{(n)} x_j^{(n)}$



Hopfield Network Convergence

- Binary probability distribution

$$P(x \mid \beta, J) = \frac{1}{Z} \exp[-\beta E(x; J)]$$

- Rewrite the expression

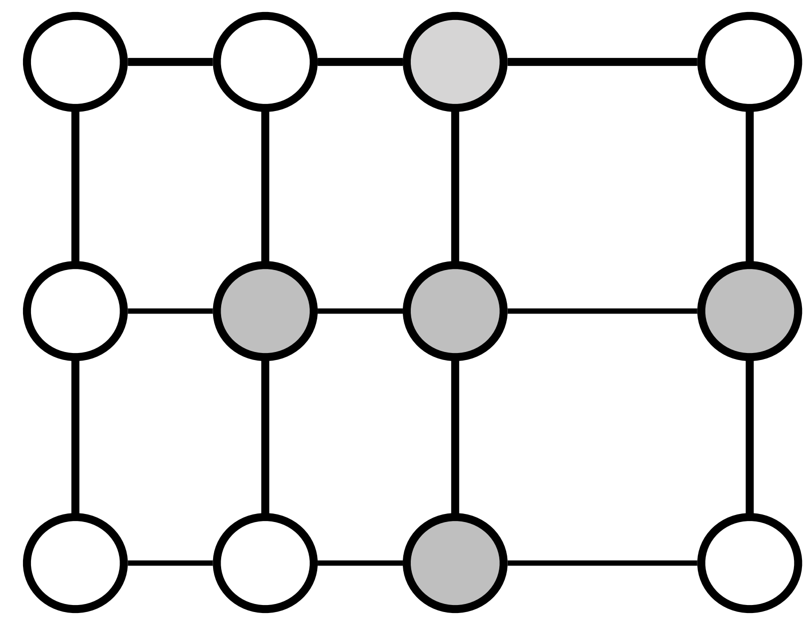
$$F(P^\star, Q) = \beta \sum_x Q(x; \theta) E(x; J) - H_Q$$

$$\equiv \beta \mathbb{E}_Q[E(x; J)] - H_Q$$

mean energy | entropy

- Energy function

$$E(x; J) = -\frac{1}{2} \sum_{m,n} J_{mn} x_m x_n - \sum_n h_n x_n$$



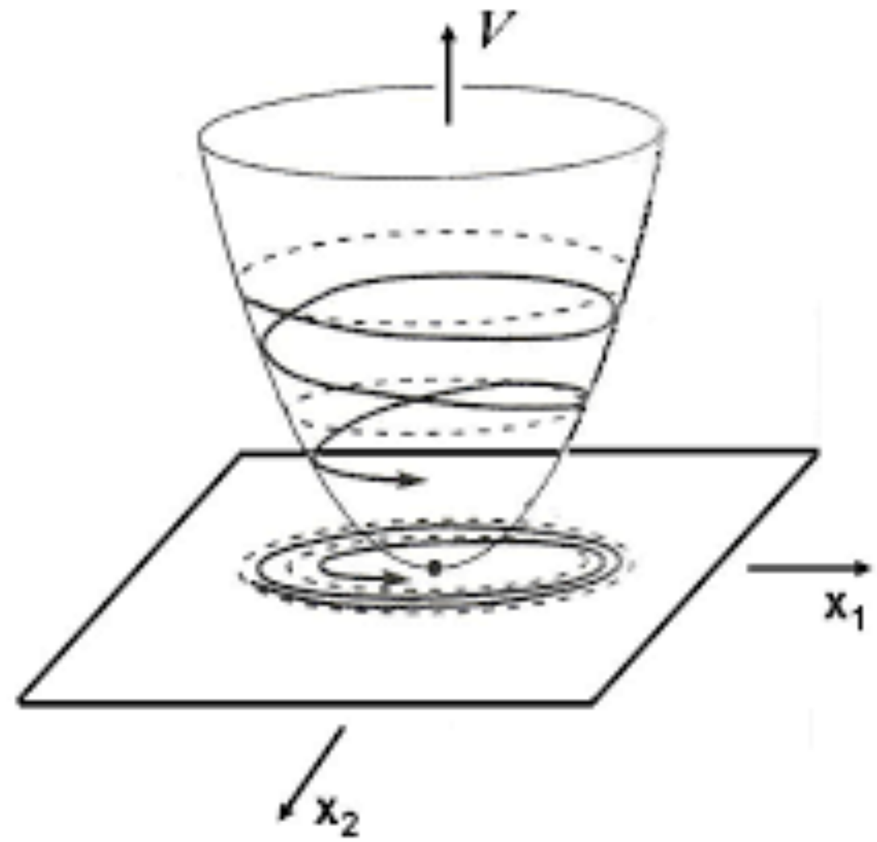
Hopfield Network Convergence

- Variational Free Energy:
$$F(P^\star, Q) = \beta\Big(-\frac{1}{2}\sum_{mn} J_{mn}\mathbb{E}_Q[x_m]\mathbb{E}_Q[x_n] - \sum_n h_n \mathbb{E}_Q[x_n]\Big) - H_Q$$

- Replace J by w, $\mathbb{E}_Q[x_n]$ by x_n, Hopfield network is identical to mean-field equations
$$F(P^\star, Q) = -\beta\frac{1}{2}\mathbf{x}^\top\mathbf{W}\mathbf{x} - \sum_i H[(1+x_n)/2] \qquad \boxed{\text{Lyapunov function}}$$

- **Stability**: Hopfield network's dynamics will always converge to a stable fixed point

Lyapunov Function

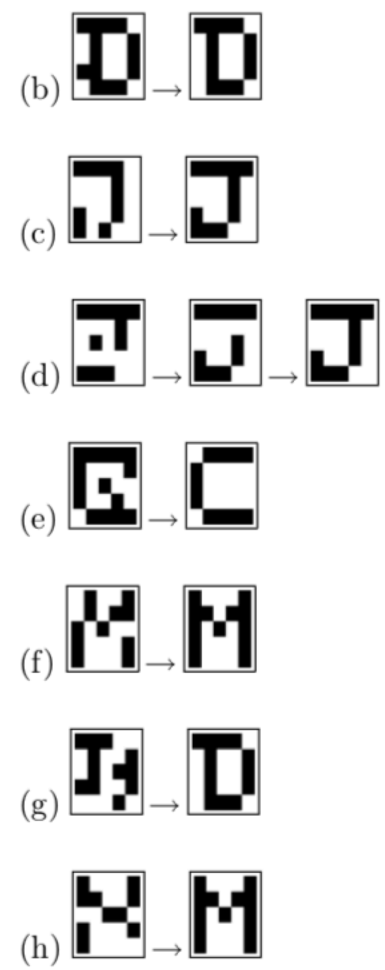


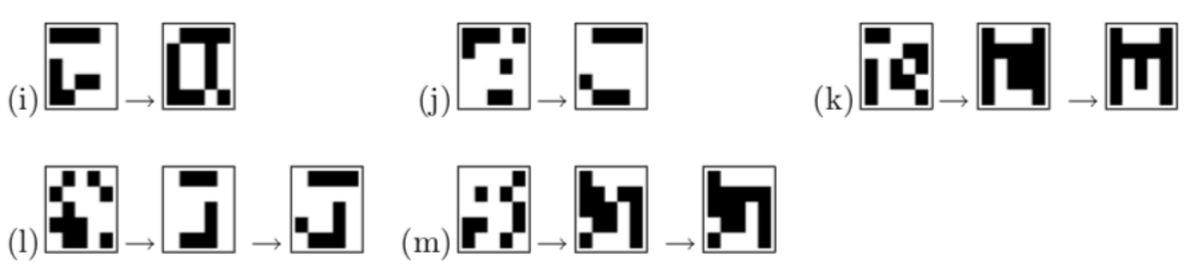
- Generalized energy function for system, characterizes the dynamics of a system

- A function V is positive definite and $\dot{V}(x) \leq 0$ for all x

Associative Memory

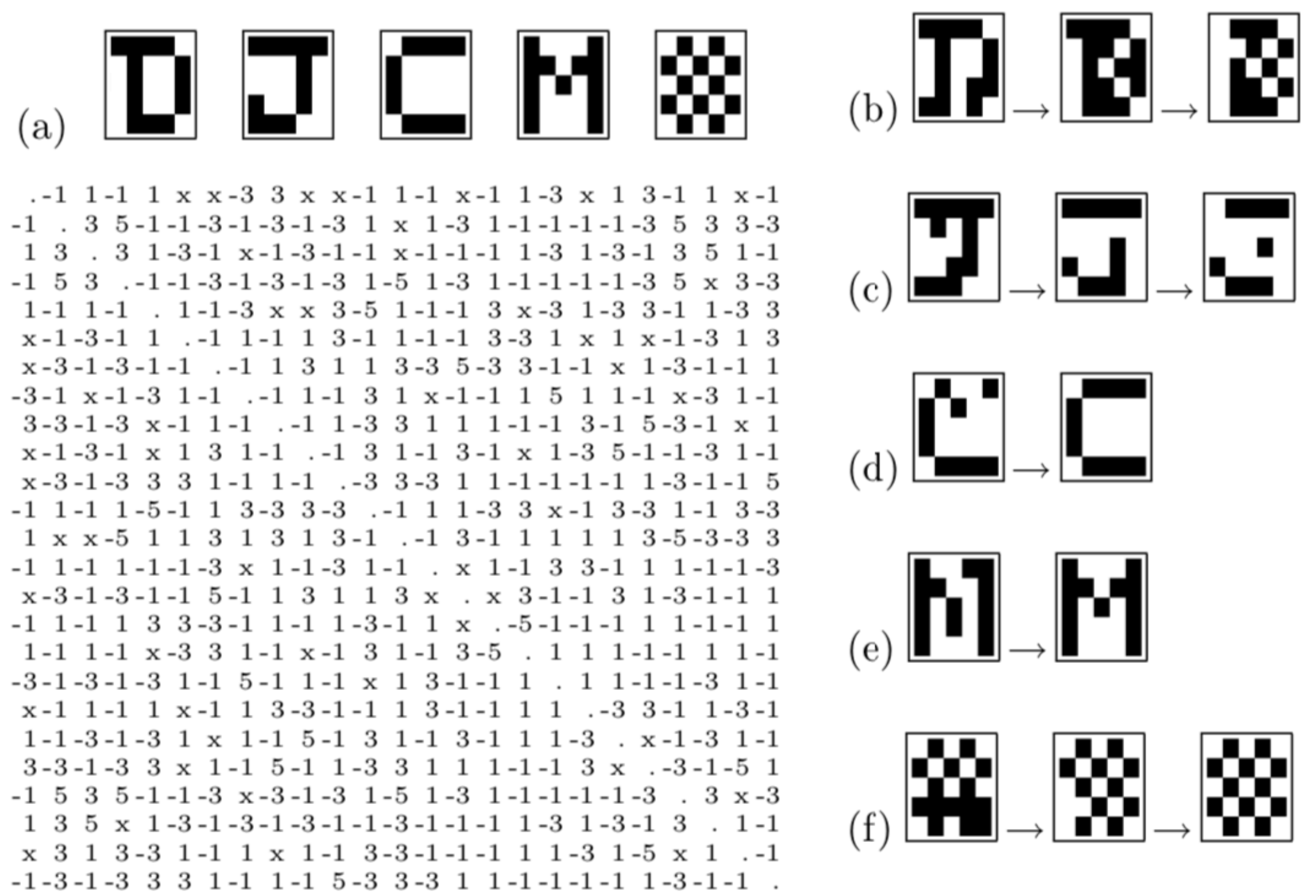
.	0	0	0	0	-2	2	-2	2	2	-2	0	0	0	2	0	0	-2	0	2	2	0	0	-2	-2
0	.	4	4	0	-2	-2	-2	-2	-2	-2	0	-4	0	-2	0	0	-2	0	-2	-2	4	4	2	-2
0	4	.	4	0	-2	-2	-2	-2	-2	-2	0	-4	0	-2	0	0	-2	0	-2	-2	4	4	2	-2
0	4	4	.	0	-2	-2	-2	-2	-2	-2	0	-4	0	-2	0	0	-2	0	-2	-2	4	4	2	-2
0	0	0	0	.	2	-2	-2	2	-2	2	-4	0	0	-2	4	-4	-2	0	-2	2	0	0	-2	2
-2	-2	-2	-2	2	.	0	0	0	0	4	-2	2	-2	0	2	-2	0	-2	0	0	-2	-2	0	4
2	-2	-2	-2	-2	0	.	0	0	4	0	2	2	-2	4	-2	2	0	-2	4	0	-2	-2	0	0
-2	-2	-2	-2	-2	0	0	.	0	0	0	2	2	2	0	-2	2	4	2	0	0	-2	-2	0	0
2	-2	-2	-2	2	0	0	0	.	0	0	-2	2	2	0	2	-2	0	2	0	4	-2	-2	-4	0
2	-2	-2	-2	-2	0	4	0	0	.	0	2	2	-2	4	-2	2	0	-2	4	0	-2	-2	0	0
-2	-2	-2	-2	2	4	0	0	0	0	.	-2	2	-2	0	2	-2	0	-2	0	0	-2	-2	0	4
0	0	0	0	-4	-2	2	2	-2	2	-2	.	0	0	2	-4	4	2	0	2	-2	0	0	2	-2
0	-4	-4	-4	0	2	2	2	2	2	2	0	.	0	2	0	0	2	0	2	2	-4	-4	-2	2
0	0	0	0	0	-2	-2	2	2	-2	-2	0	0	.	-2	0	0	2	4	-2	2	0	0	-2	-2
2	-2	-2	-2	-2	0	4	0	0	4	0	2	2	-2	.	-2	2	0	-2	4	0	-2	-2	0	0
0	0	0	0	4	2	-2	-2	2	-2	2	-4	0	0	-2	.	-4	-2	0	-2	2	0	0	-2	2
0	0	0	0	-4	-2	2	2	-2	2	-2	4	0	0	2	-4	.	2	0	2	-2	0	0	2	-2
-2	-2	-2	-2	-2	0	0	4	0	0	0	2	2	2	0	-2	2	.	2	0	0	-2	-2	0	0
0	0	0	0	0	-2	-2	2	2	-2	-2	0	0	4	-2	0	0	2	.	-2	2	0	0	-2	-2
2	-2	-2	-2	-2	0	4	0	0	4	0	2	2	-2	4	-2	2	0	-2	.	0	-2	-2	0	0
2	-2	-2	-2	2	0	0	0	4	0	0	-2	2	2	0	2	-2	0	2	0	.	-2	-2	-4	0
0	4	4	4	0	-2	-2	-2	-2	-2	-2	0	-4	0	-2	0	0	-2	0	-2	-2	.	4	2	-2
0	4	4	4	0	-2	-2	-2	-2	-2	-2	0	-4	0	-2	0	0	-2	0	-2	-2	4	.	2	-2
-2	2	2	2	-2	0	0	0	-4	0	0	2	-2	-2	0	-2	2	0	-2	0	-4	2	2	.	0
-2	-2	-2	-2	2	4	0	0	0	0	4	-2	2	-2	0	2	-2	0	-2	0	0	-2	-2	0	.



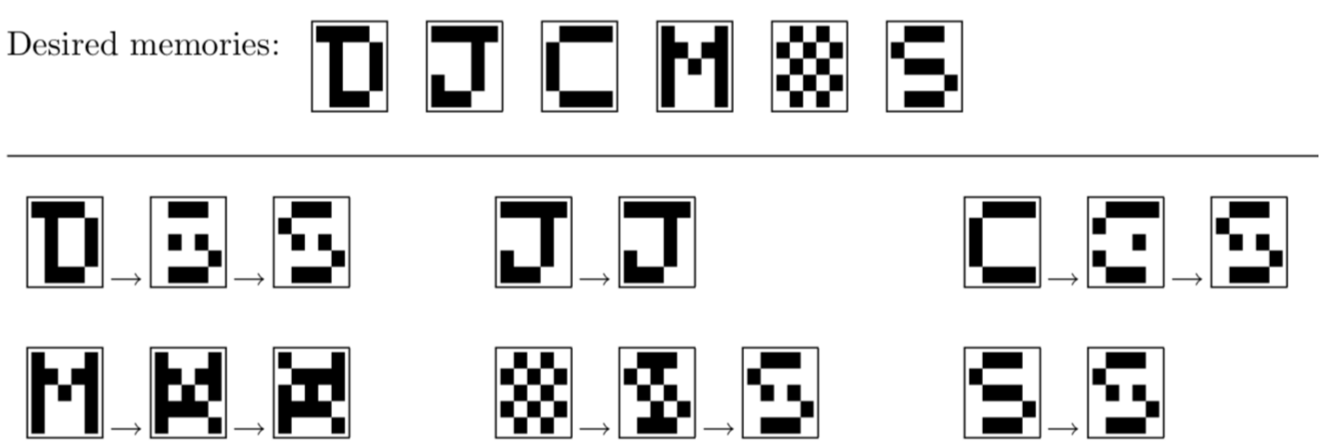


- 25 -unit binary Hopfield network
- (a) four patterns as 5x5 binary images
- (b)- (m) evolution of state of the network

Robustness



- randomly delete 26 weights (x)
- (a) five memories
- (b-f) initial states: some are restored, some converge to other states

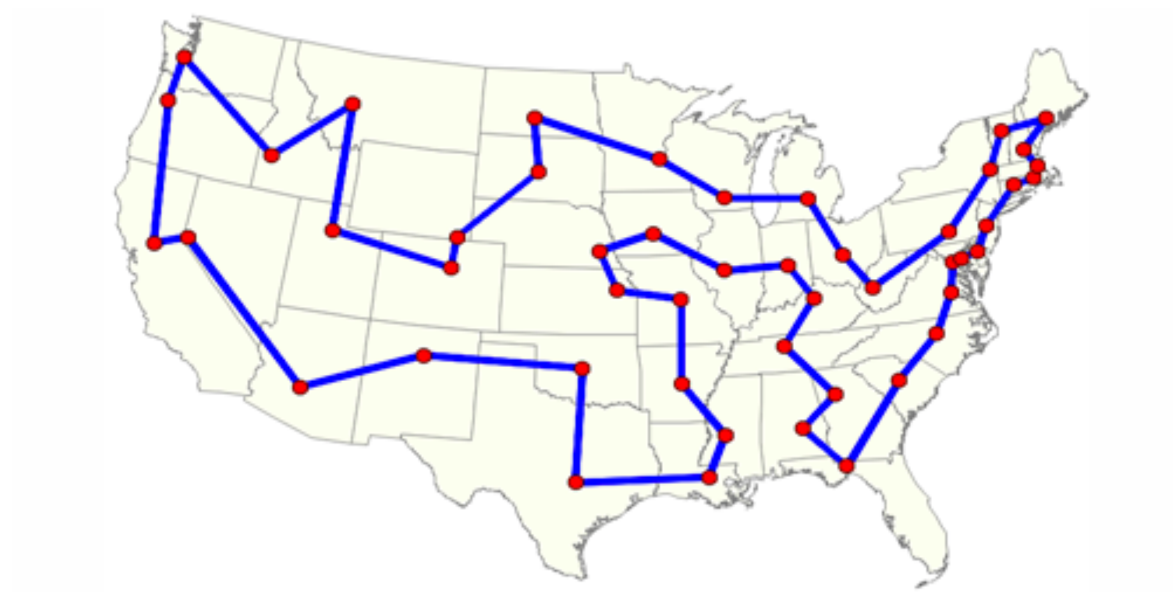


- Overloaded network fails catastrophically

Continuous Hopfield Network

- Each neuron's activity x_i is a continuous function of time, activations $a_i(t) = \sum_j w_{ij} x_j(t)$
- Neuron's response mediated by the differential equation: $\dfrac{d}{dt} x_i(t) = -\dfrac{1}{\tau}(x_i(t) - f(a_i))$ where $f(a)$ is the activation function
- The weight matrix is symmetric —> variational free energy is the Lyapunov function

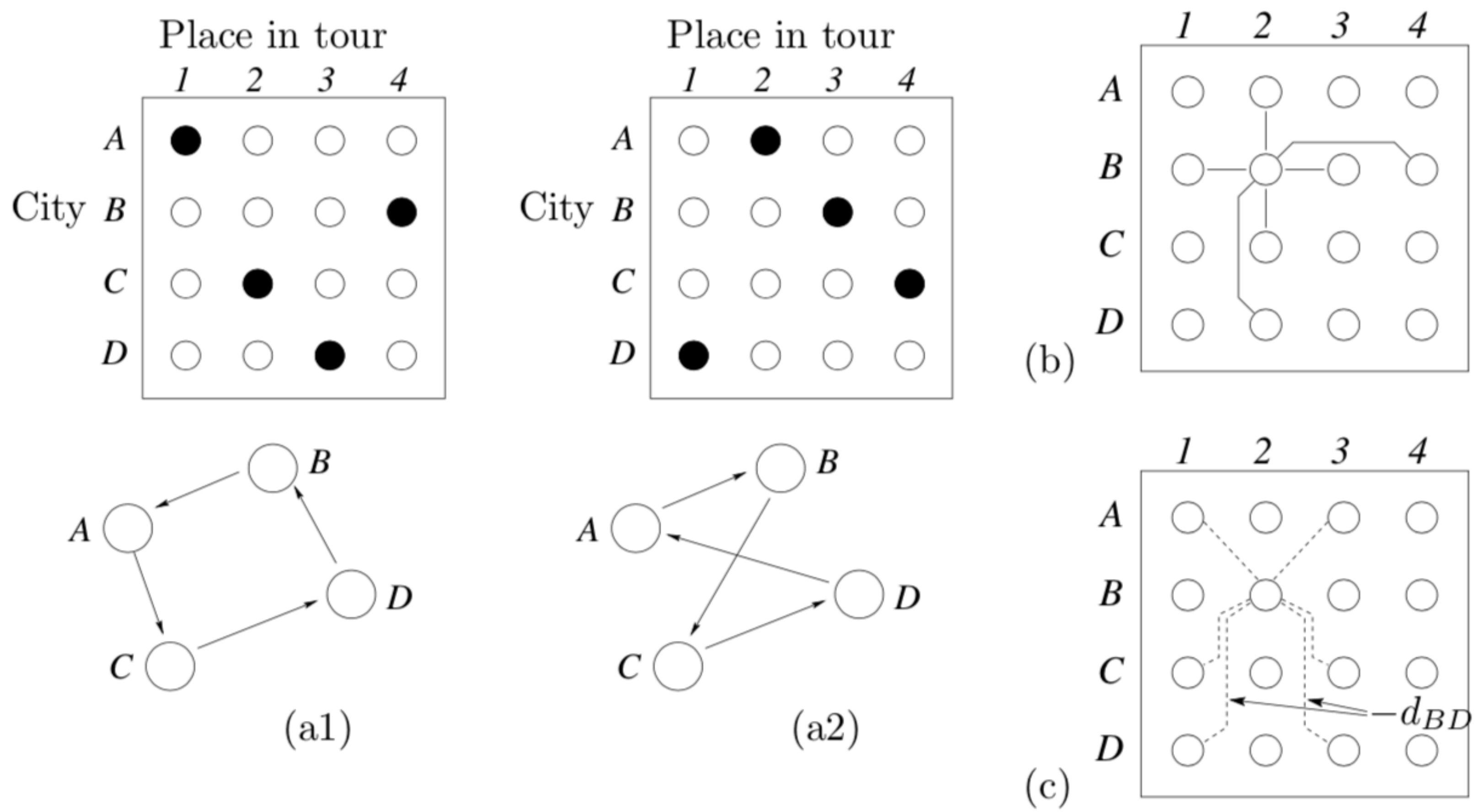
Solving Optimization Problems



- Map interesting optimization problems onto Hopfield networks: *constrained satisfaction*

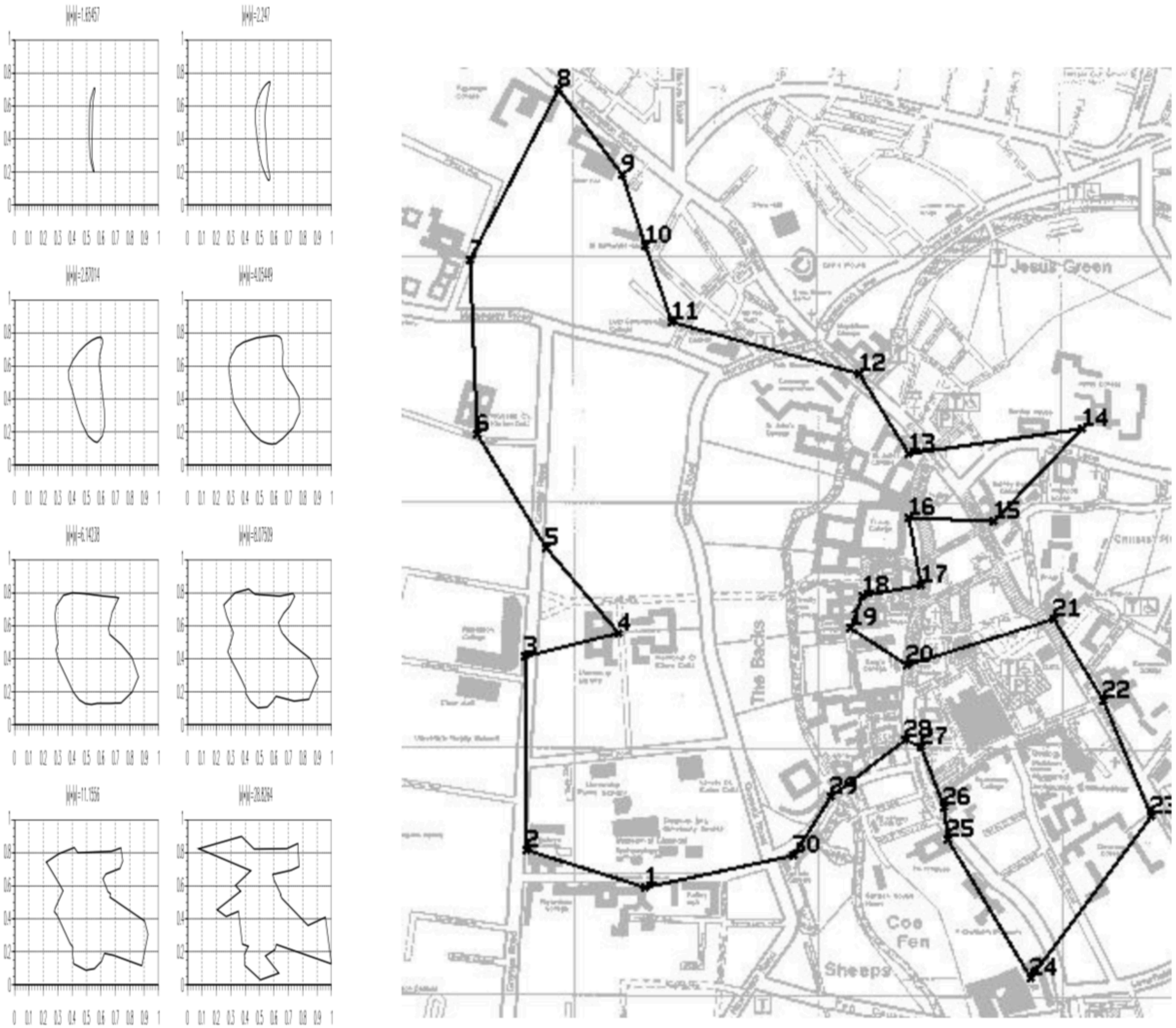
- e.g. Solving a traveling salesman problem with K cities
  - A set of K cities, a matrix of K(K-2)/2 distances between cities
  - Find a closed tour of the cities, visiting each city once, has the smallest total distance

TSP on Hopfield Network



- **States have exactly one `1` in every row/column**: putting large negative weights between any pair of neurons in the same row/column
- **Weights encode the total distance**: putting negative weights proportional to the distance between the nodes

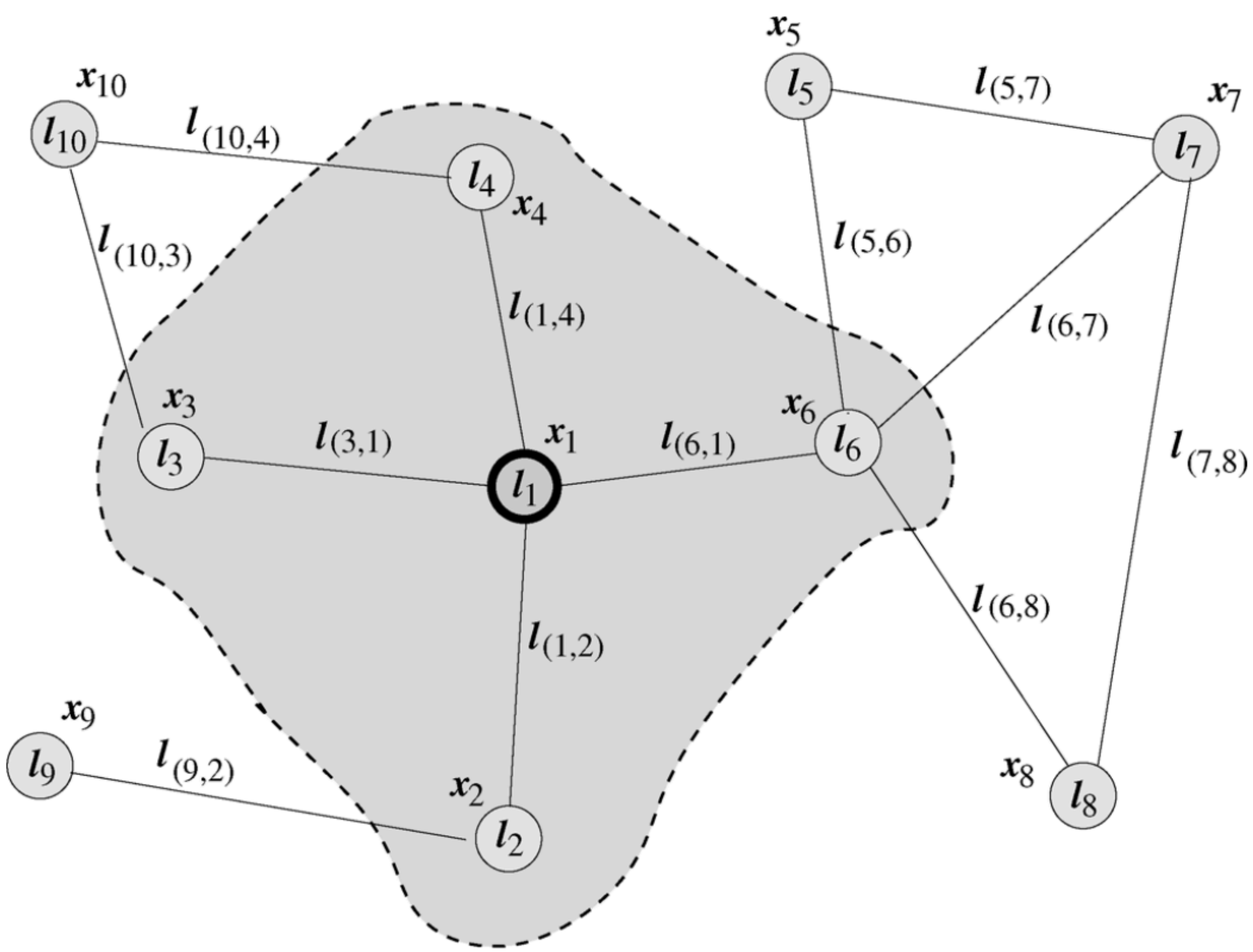
TSP on Hopfield Network



(a) (b)

- Evolution of continuous Hopfield network
- Shortest path linking 27 Cambridge colleges

Graph Neural Networks

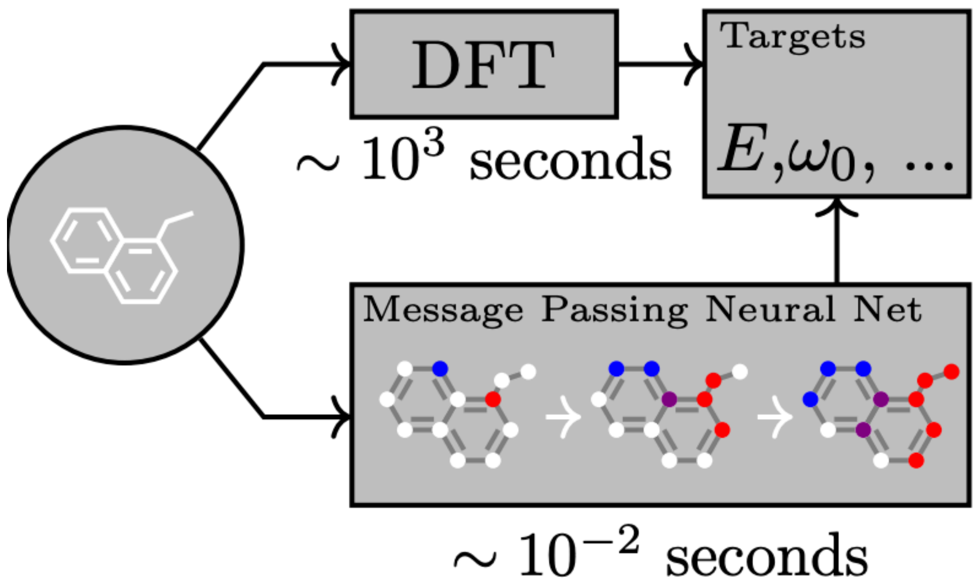


$$x_1 = f_w(l_1, \underbrace{l_{(1,2)}, l_{(3,1)}, l_{(1,4)}, l_{(6,1)}}_{l_{co[1]}}, \underbrace{x_2, x_3, x_4, x_6}_{x_{ne[1]}}, \underbrace{l_2, l_3, l_4, l_6}_{l_{ne[n]}})$$

- State of a node depends on its neighbors
  $$x_i = \sum_{j \in \mathcal{N}(i)} f(l_i, l_{i,j}, x_j, l_j)$$
  where l is the label for nodes/edges
- Optimize by minimizing the energy

Scarselli, Franco, et al. "The graph neural network model." *IEEE Transactions on Neural Networks* 20.1 (2008): 61-80.

Neural Message Passing



- A graph G with node features x_v and edge features e_{vw}

- Message passing for T steps

$$m_v^{t+1} = \sum_{w \in \mathcal{N}(v)} M_t(h_v^t, h_w^t, e_{vw}) \quad h_v^{t+1} = U_t(h_v^t, m_v^{t+1}) \quad \hat{y} = R(\{h_v\})$$

Message function M_t Update function U_t and Readout function R are all neural networks

Gilmer, Justin, et al. "Neural message passing for quantum chemistry." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.