Distributed Optimization

MAHI 2013 Workshop

Richard Heusdens

October 14, 2013

Signal and Information Processing Lab

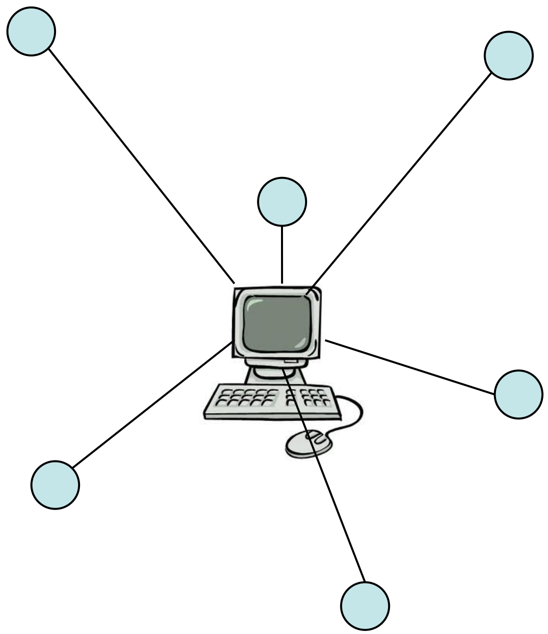
Delft University of Technology

Distributed Signal Processing

Due to the explosion in size and complexity of modern datasets (Big Data), it is increasingly important to be able to solve problems with a very large number of features or training examples. Hence, it is either necessary or at least highly desirable to have

- decentralized collection or storage of these datasets
- distributed solution for the problems

Introduction



Example: $Ax = b \;\Rightarrow\; x = A^{-1}b$

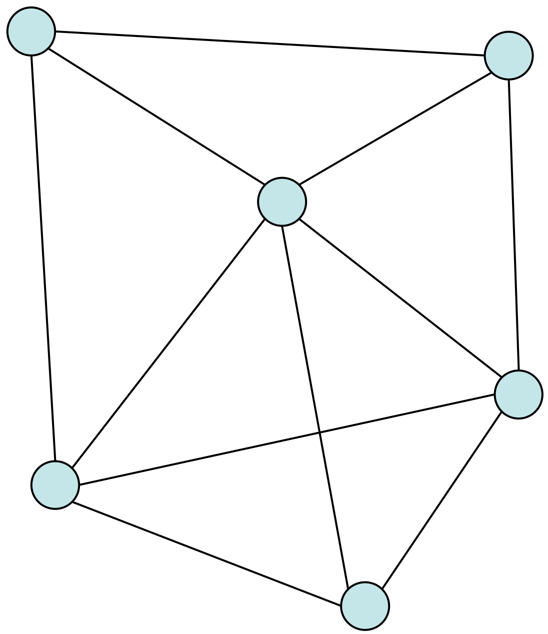
- Centralized computing
- Not well scalable
- Sensitive to sensor failures
- Single point of failure

Introduction

New sensor concepts:

- connecting a large number of small and inexpensive sensors in a *sensor network*
- building blocks have a sensing component and limited data-processing and communication power
- de-centralized (*distributed*) processing

Introduction



How to compute $x = A^{-1}b$?

- Well scalable
- Robust against sensor failures
- Independent of network topology

Distributed Signal Processing

Possible solutions:

- convex optimization (alternating direction method of multipliers (ADMM)
- probabilistic inference (maximum a posteriori (MAP) probabilities)

Contents

PART I:

- Convex optimization
- Dual ascent/method of multipliers
- Alternating direction method of multipliers (ADMM)

PART II:

- Graphical models
- Probabilistic inference
- Message passing

PART I

Convex Optimization

Convex Optimization

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p \end{array}$$

optimal value:

$$p^* = \inf\{f_0(x) \mid f_i(x) \leq 0,\ i = 1, \ldots, m,\ h_i(x) = 0,\ i = 1, \ldots, p\}$$

x is feasible if $x \in \mathbf{dom}\, f_0$ and it satisfies the constraints.
x is optimal if $f_0(x) = p^*$.

Lagrange Dual Function

Lagrangian:

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x)$$

Lagrange dual function:

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) \leq p^*$$

proof: ($\tilde{x}$ feasible)

$$f_0(\tilde{x}) \geq L(\tilde{x}, \lambda, \nu) \geq \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = g(\lambda, \nu)$$

Lagrange Dual Problem

Lagrange dual problem:

$$\begin{array}{ll} \text{maximize} & g(\lambda, \nu) \\ \text{subject to} & \lambda \succeq 0 \end{array}$$

strong duality: $(d^* = \sup\{g(\lambda, \nu) | \lambda \succeq 0\})$

For convex functions we (usually) have $d^* = p^*$

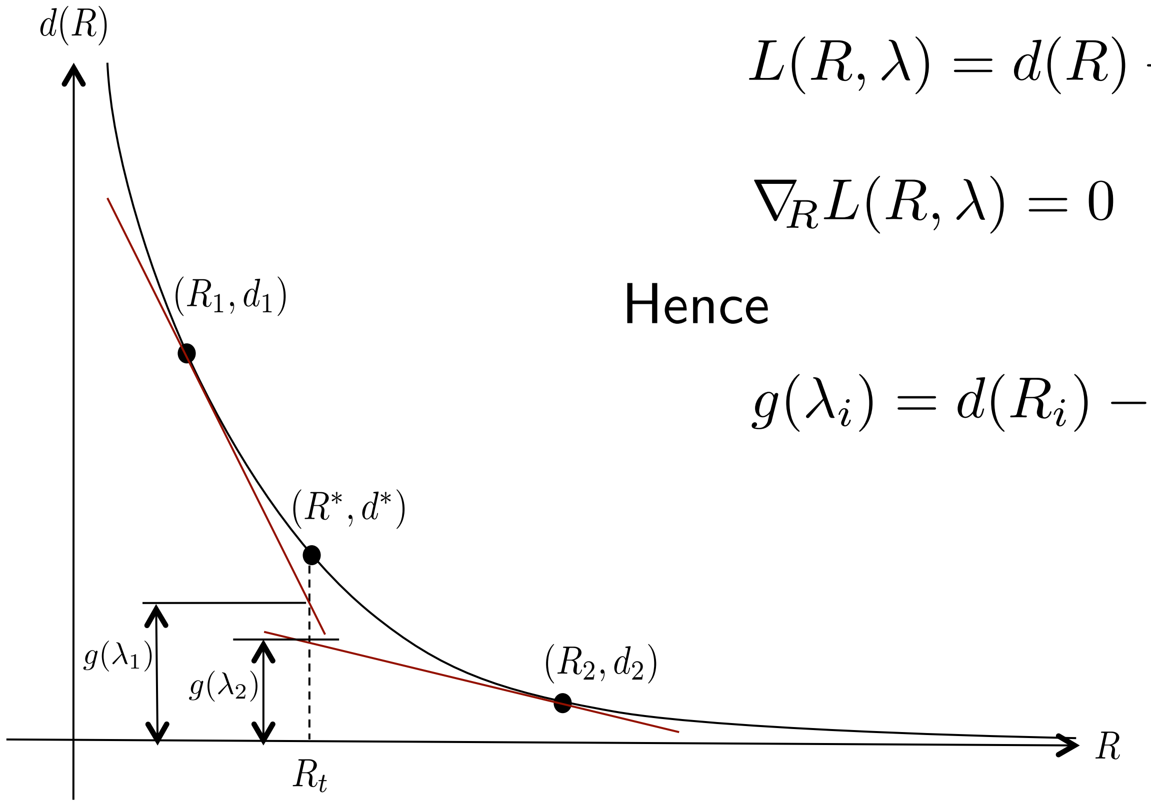
Example: Source Coding

Source coding problem:

Let R denote the rate (number of bits) to represent $X \in \mathcal{X}$ by $\hat{X} \in \hat{\mathcal{X}}$. Given a distortion measure $d : \mathcal{X} \times \hat{\mathcal{X}} \mapsto \mathbb{R}^+$:

$$\begin{array}{ll} \text{minimize} & d(R) \\ \text{subject to} & R \leq R_t \end{array}$$

Example: Source Coding



$$L(R,\lambda) = d(R) + \lambda(R - R_t)$$

$$\nabla_R L(R,\lambda) = 0 \quad \Rightarrow \quad \lambda = -d'(R)$$

Hence

$$g(\lambda_i) = d(R_i) - d'(R_i)(R_i - R_t)$$

Dual Ascent

Consider the equality-constrained convex optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b \end{array}$$

The Lagrangian is given by

$$L(x, \nu) = f(x) + \nu^t (Ax - b)$$

We can recover a primal optimal point x^* from a dual optimal point ν^* as

$$x^* = \arg\min_x L(x, \nu^*)$$

Dual Ascent

In the dual ascent method, we solve the dual problem using gradient ascent. If $x^+ = \arg\min_x L(x, \nu)$, and thus $g(\nu) = L(x^+, \nu)$, we have

$$\nabla g(\nu) = Ax^+ - b$$

Dual ascent algorithm:

$$x^{k+1} := \arg\min_x L(x, \nu^k)$$
$$\nu^{k+1} := \nu^k + \alpha^k (Ax^{k+1} - b)$$

where $\alpha^k > 0$ is a step size.

Dual Decomposition

Suppose f is separable:

$$f(x) = f_1(x_1) + f_2(x_2) + \cdots + f_N(x_N), \quad x = (x_1, x_2, \ldots, x_N)$$

then $L(x, \nu)$ is separable and the dual ascent splits into N separate minimization

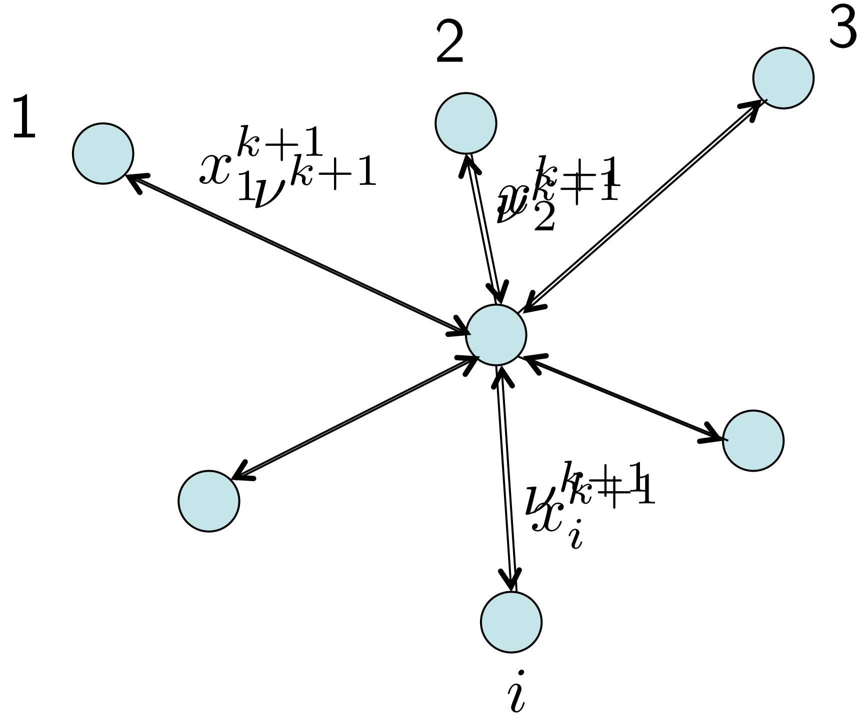
$$x_i^{k+1} := \arg\min_{x_i} L_i(x_i, \nu^k)$$

which can be carried out in parallel!

Dual Decomposition

step 1 (broadcast):

step 2 (gather):



Method of Multipliers

Augmented Lagrangian;

$$L_\rho(x, \nu) = f(x) + \nu^t(Ax - b) + \rho/2\|Ax - b\|_2^2$$

where the penalty function ($\rho > 0$) is introduced to bring robustness to the dual ascent method.

The augmented Lagrangian can be viewed as the (unaugmented) Lagrangian associated with

$$\begin{array}{ll} \text{minimize} & f(x) + \rho/2\|Ax - b\|_2^2 \\ \text{subject to} & Ax = b \end{array}$$

Method of Multipliers

Method of multipliers:

$$x^{k+1} := \arg\min_x L_\rho(x, \nu^k)$$
$$\nu^{k+1} := \nu^k + \rho(Ax^{k+1} - b)$$

Since
$$\begin{aligned} 0 &= \nabla_x L_\rho(x^{k+1}, \nu^k) \\ &= \nabla_x f(x^{k+1}) + A^t(\nu^k + \rho(Ax^{k+1} - b)) \\ &= \nabla_x f(x^{k+1}) + A^t \nu^{k+1} \\ &= \nabla_x L_0(x^{k+1}, \nu^{k+1}) \end{aligned}$$

the iterate (x^{k+1}, ν^{k+1}) is dual feasible!

Method of Multipliers

+ convergence under much more relaxed conditions (f can be non differentiable, take on value $+\infty$, ...)

– but the quadratic penalty function destroys splitting of the x-update

$$L_\rho(x,\nu) = f(x) + \nu^t(Ax - b) + \rho/2\|Ax - b\|_2^2$$

idea: decouple the primal constraints by introducing a new variable

ADMM

ADMM problem: minimize $f(x) + g(z)$

subject to $Ax + Bz = c$

The augmented Lagrangian is given by

$$L_\rho(x, z, \nu) = f(x) + g(z) + \nu^t(Ax + Bz - c) + \rho/2\|Ax + Bz - c\|_2^2$$

ADMM

$$\begin{aligned} x^{k+1} &:= \arg\min_x L_\rho(x, z^k, \nu^k) \\ z^{k+1} &:= \arg\min_z L_\rho(x^{k+1}, z, \nu^k) \\ \nu^{k+1} &:= \nu^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{aligned}$$

Convergence

- f closed, proper, and convex
- L_0 has a saddle point

Under these assumptions, the ADMM iterates satisfy:

- *Residual convergence:* $Ax^k + Bz^k - c \to 0$
- *Objective convergence:* $f(x^k) + g(z^k) \to p^*$
- *Dual variable convergence:* $\nu^k \to \nu^*$

Example: Consensus

Consider the problem

$$
\begin{array}{ll}
\text{minimize} & f(x) = \sum_{i=1}^{N} f_i(x) \\
\text{subject to} & x_i = x_j \text{ for all } (i, j)
\end{array}
$$

This problem can be rewritten with local variables x_i and a common global variable z

$$
\begin{array}{ll}
\text{minimize} & \sum_{i=1}^{N} f_i(x_i) + g(z) \\
\text{subject to} & x_i = z,\ i - 1, \ldots, N
\end{array}
$$

Example: Consensus

ADMM

$$x_i^{k+1} := \arg\min_{x_i} \left(f_i(x_i) + {\nu_i^k}^t (x_i - z^k) + \rho/2 \|x_i - z^k\|_2^2 \right)$$

$$z^{k+1} := \arg\min_{z} \sum_{i=1}^{N} \left({\nu_i^k}^t (x_i^{k+1} - z) + \rho/2 \|x_i^{k+1} - z\|_2^2 \right)$$

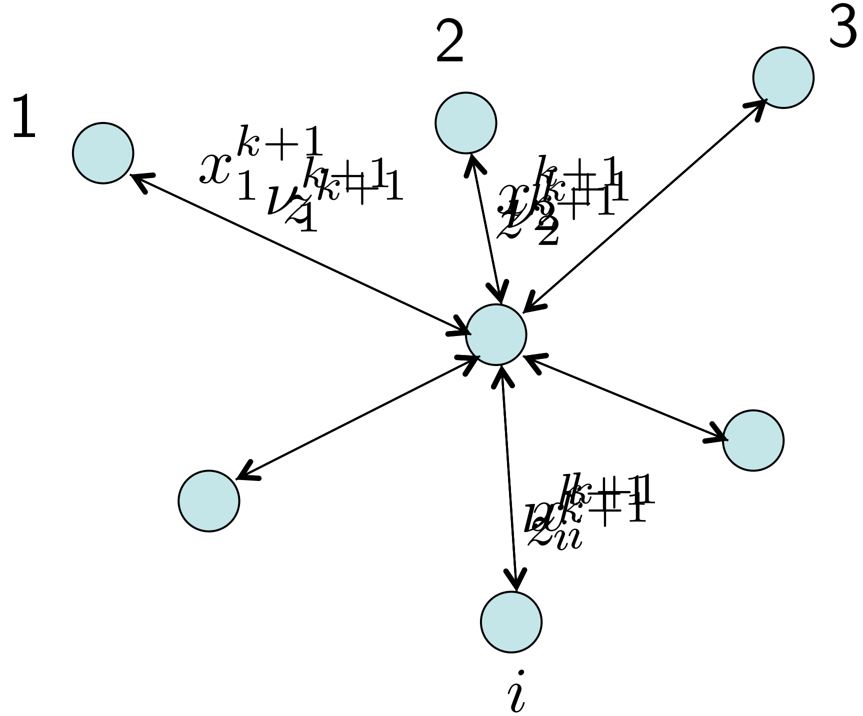
$$= \frac{1}{N} \sum_{i=1}^{N} (1/\rho \nu_i^k + x_i^{k+1})$$

$$\nu_i^{k+1} := \nu_i^k + \rho(x_i^{k+1} - z^{k+1})$$

ADMM

step 1 (broadcast):

step 2 (gather):



PART II

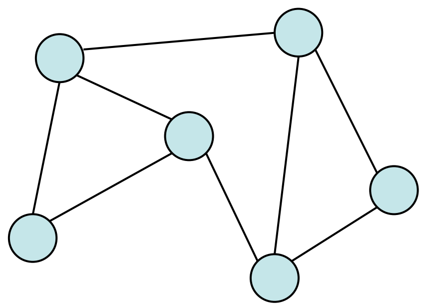
Probabilistic Inference

Graphical Models

- Efficiently represent a joint distribution over a set of random variables, each represented by a node in a graph
- Even in the simplest case where these variables are binary-valued, a joint distribution requires the specification of 2^n numbers
- If there is some structure in the distribution, we can factor the distribution into modular components.
- The structure that graphical models exploit is the independence properties that exist in many real-world phenomena.

Graphical Models

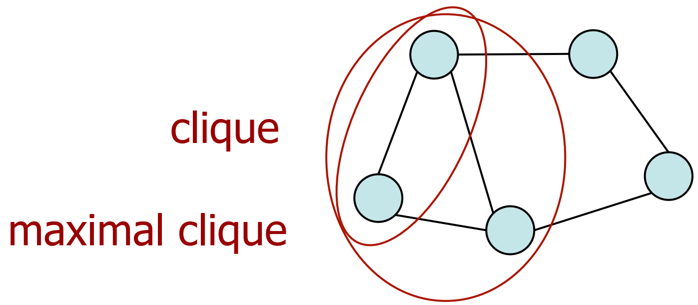
- Typically, a graph $G = (V, E)$ is depicted in diagrammatic form as a set of dots for the vertices, joined by lines for the edges.



- A graph is said to be *acyclic* if it is a graph without cycles.
- A *tree* is a graph in which there is one, and only one, path between any pair of nodes. As a consequence, trees are acyclic.

Graphical Models

- A node i has *neighbors* $\mathcal{N}(i) = \{j \in V : (i,j) \in E\}$.
- A *clique* is defined as a subset which is fully connected.
- A *maximal clique* is a clique such that it is not possible to include any other node from the graph in the set without it ceasing to be a clique.



Graphical Models

The joint distribution factorizes as a product of *potential functions* over all, say m, maximal cliques of the graph

$$p(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{i=1}^{m} \psi_i(x_{C_i}),$$

where $\psi_i(x_{C_i}) \geq 0$ and

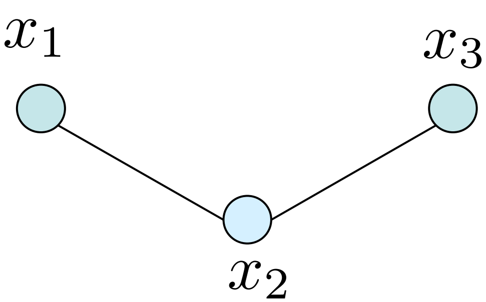
$$Z = \sum_{X_1, \ldots, X_n} p(x_1, \ldots, x_n).$$

Markov Random Field

Example:

Consider the random variable X_1, X_2 and X_3 and assume that we know that X_1 is conditionally independent of X_3 given X_2.

Graph:



We then have

$$p(x_1, x_2, x_3) = \frac{1}{Z}\psi_1(x_1, x_2)\psi_2(x_2, x_3).$$

$$\Rightarrow \quad p(x_1|x_2, x_3) = p(x_1|x_2)$$

Probabilistic Inference

There are two basic kinds of *inference* problems that often arise:

- marginal probabilities:

$$p(x_F) = \sum_{x_G} p(x_F, x_G).$$

- maximum a posteriori (MAP) probabilities;

$$p^*(x_F) = \max_{x_G} p(x_F, x_G).$$

Probabilistic Inference

Why are we interested in computing, e.g., MAP probabilities?

Suppose we want to solve $Ax = b$ for a symmetric matrix A (e.g. a correlation matrix). We then can construct the quadratic function

$$q(x) = \frac{1}{2}x^t Ax - b^t x.$$

Then equating $\partial q/\partial x = 0$ gives the stationary point x^* which is the solution to $Ax = b$.

Probabilistic Inference

Let us define the joint Gaussian distribution

x^*

$$p(x) = \frac{1}{Z} e^{-\frac{1}{2}x^t A x + b^t x} = \mathcal{N}\left(A^{-1}b, A^{-1}\right),$$

which we can factorize into a product of potential functions.

Hence we have

$$x^* = \arg\max_x p(x)$$

Probabilistic Inference

Similarly, we can solve

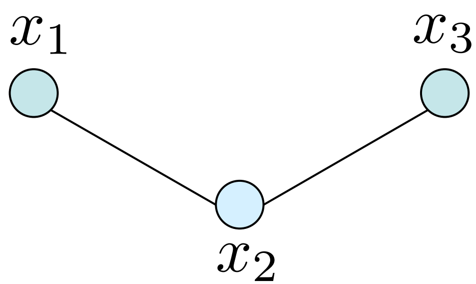
$$\min_x \|Ax - b\|^2$$

for $A \in \mathbb{C}^{n \times k}, n \geq k$, of full rank. The optimal solution is given by

$$x^* = (A^t A)^{-1} A^t b = J^{-1} h$$

Markov Random Field

Example (con't)



Assume we want tot compute the MAP distribution $p^*(x_2)$. Direct computation yields

$$p^*(x_2) = \max_{x_1} \max_{x_3} p(x_1, x_2, x_3) \qquad \mathcal{O}(s^n)$$

Using the factorization over maximal cliques, we obtain

$$p^*(x_2) = \frac{1}{Z} \max_{x_1} \psi_1(x_1, x_2) \max_{x_3} \psi_2(x_2, x_3) \quad \mathcal{O}(ns^2)$$

Probabilistic Inference

In practice, products of small probabilities can lead to numerical problems, and so it is convenient to work with the logarithm of the joint distribution. Taking the logarithm simply has the effect of replacing products by sums

$$\begin{aligned}\ln p(x1, \ldots, x_n) &= \sum_{i=1}^{m} \ln \psi_i(x_{C_i}) \\ &= \sum_{i=1}^{m} f_i(x_{C_i})\end{aligned}$$

Probabilistic Inference

Consider the quadratic optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \left(\frac{1}{2} x^t J x - h^t x \right)$$

To achieve this goal, we decompose $f(x)$ in a pairwise fashion according to a graph $G = (V, E)$, so that

$$f(x) = \sum_{i \in V} f_i(x_i) + \sum_{(i,j) \in E} f_{i,j}(x_i, x_j)$$

Probabilistic Inference

The local functions are given by (assuming J has unit diagonal elements)

$$f_i(x_i) = \frac{1}{2}x_i^2 - h_i x_i, \quad i \in V$$

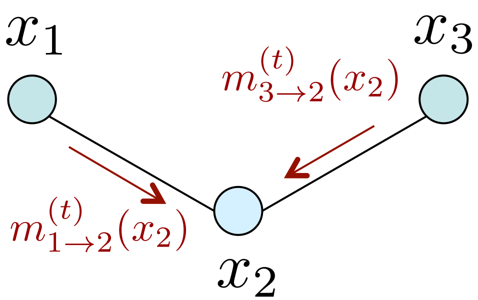
$$f_{i,j}(x_i, x_j) = J_{ij} x_i x_j, \qquad (i, j) \in E$$

Message Passing

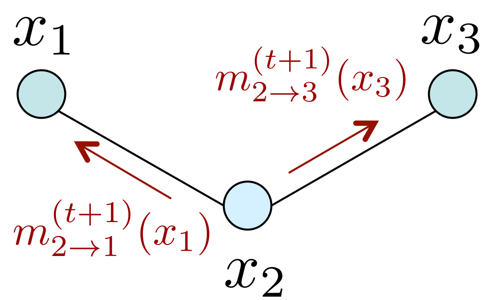
A message passing algorithm exchanges information between nodes iteratively until reaching consensus.

In particular, at time t, each node i collects incoming messages $\{m_{u\to i}^{(t)}(x_i)|u \in \mathcal{N}(i)\}$ from all neighboring nodes. These messages are then combined to produce new outgoing messages, one for each neighbor $u \in \mathcal{N}(i)$.

time t:



time $t+1$:



Message Passing

At each time t each node i computes an estimate $\hat{x}_i^{(t)}$ of the optimal solution x_i^* by minimizing the self potential

$$\hat{x}_i^{(t)} = \arg\min_{x_i} \left(f_i(x_i) + \sum_{u \in \mathcal{N}(i)} m_{u \to i}^{(t)}(x_i) \right), \quad i \in V$$

If the algorithm converges to the optimal solution, we have

$$\lim_{t \to \infty} \hat{x}^{(t)} = x^*.$$

Min-Sum Algorithm

The key problem in message-passing algorithms is how to define the updating expressions for $m^{(t)}_{j\to i}(x_i)$

min-sum algorithm:

$$m^{(t)}_{i\to j}(x_j) = \min_{x_i}\left(f_i(x_i) + f_{ij}(x_i, x_j) + \sum_{u\in\mathcal{N}(i)\setminus j} m^{(t-1)}_{u\to i}(x_i)\right)$$
$$= \gamma^{(t)}_{ij} x_j^2 + z^{(t)}_{ij} x_j$$

Message Passing

(generalized) linear coordinate-descent (LiCD) algorithm:

$$m_{i\to j}^{(t)}(x_j) = z_{ij}^{(t)} x_j$$

[1] G. Zhang and R. Heusdens. Linear Coordinate-Descent Message-Passing for Quadratic Optimization. *Neural Computation*. 2012.

[2] G. Zhang and R. Heusdens. Generalized Linear Coordinate-Descent Message-Passing for Convex Optimization. *International Conference on Acoustics, Speech and Signal Processing*. pp. 2009-2012, 2012.

Conclusions (1)

- Distributed optimization through convex optimization or probabilistic inference
- Alternating direction method of multipliers combines the decomposability of dual ascent and the robustness of the method of multipliers
- key problem is to re-formulate the original problem into one that matches ADMM

Conclusions (2)

- Graphical models can be used to exploit structure in the problem
- Key problem in message passing is the design of the messages (convergence, computational complexity, transmission power, etc.)