Searching for Scalable Solutions to really Large Problems: Theory

Luis Lafuente and Scott Kirkpatrick
Center for Bits and Atoms
Massachusetts Institute of Technology

Optimization theory as a modeling tool

- Decoding over noisy channels
- Optimal routing
- Scheduling
- Resource allocation
- Solving Linear Systems
- MAP inference

Can be formulated as a (**combinatorial**) **optimization problem**

Optimization Theory and Convexity

"...the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity."

R. Tyrrell Rockafellar (SIAM Review, 2003)

Lagrange Duality

Primal

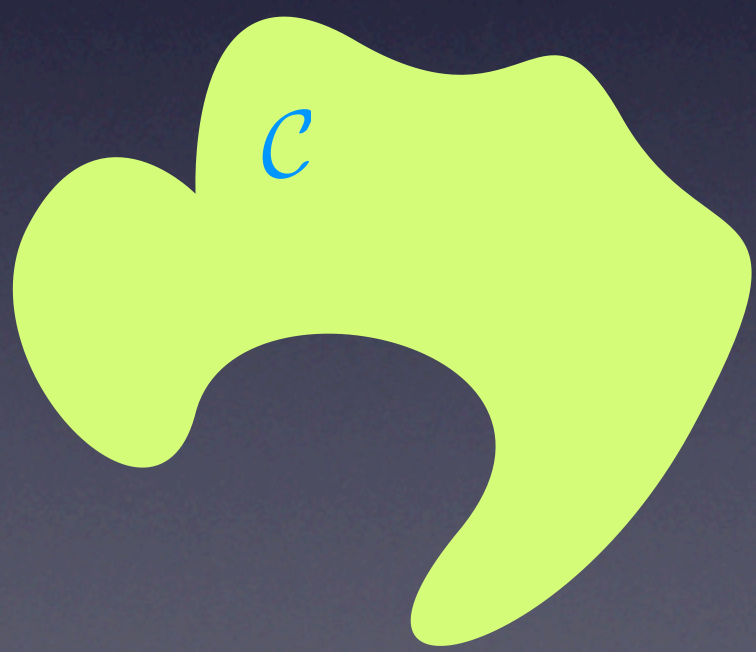
$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0 \quad 1 \leq i \leq m, \\ & h_i(\mathbf{x}) = 0 \quad 1 \leq i \leq p \end{aligned}$$

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

Dual

$$\begin{aligned} \underset{\boldsymbol{\lambda}, \boldsymbol{\nu}}{\text{maximize}} \quad & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned}$$

Convex Relaxations



Optimization Theory and Convexity

"...the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity."

R. Tyrrell Rockafellar (SIAM Review, 2003)

Lagrange Duality

Primal

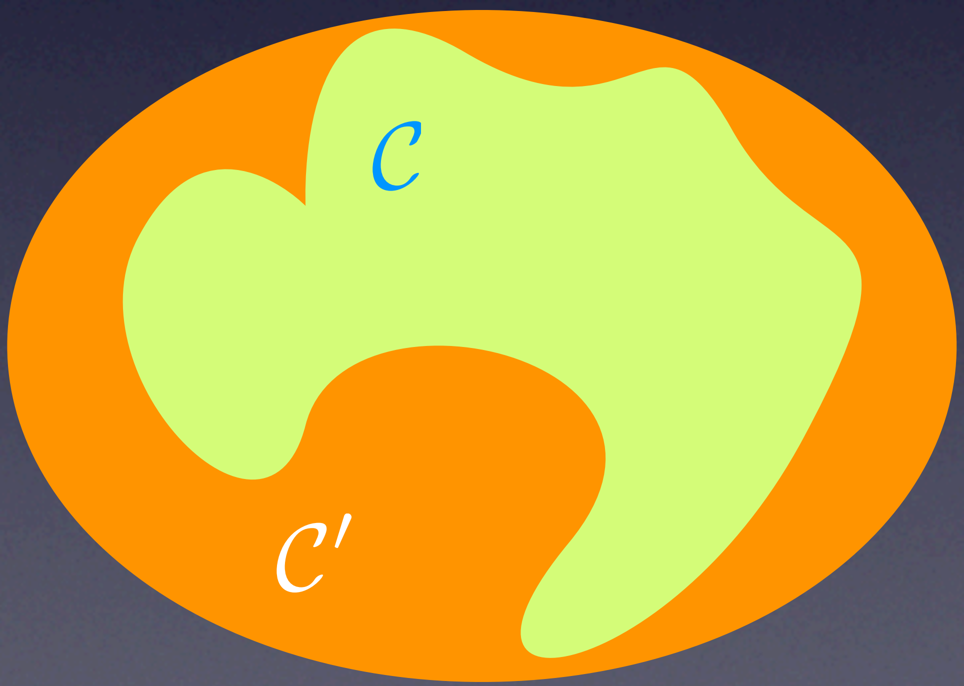
$$\begin{array}{lll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) & \\ \text{subject to} & f_i(\mathbf{x}) \leq 0 & 1 \leq i \leq m, \\ & h_i(\mathbf{x}) = 0 & 1 \leq i \leq p \end{array}$$

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

Dual

$$\begin{array}{ll} \underset{\boldsymbol{\lambda}, \boldsymbol{\nu}}{\text{maximize}} & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{subject to} & \boldsymbol{\lambda} \geq 0 \end{array}$$

Convex Relaxations



Optimization Theory and Convexity

"...the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity."

R. Tyrrell Rockafellar (SIAM Review, 2003)

Lagrange Duality

Primal

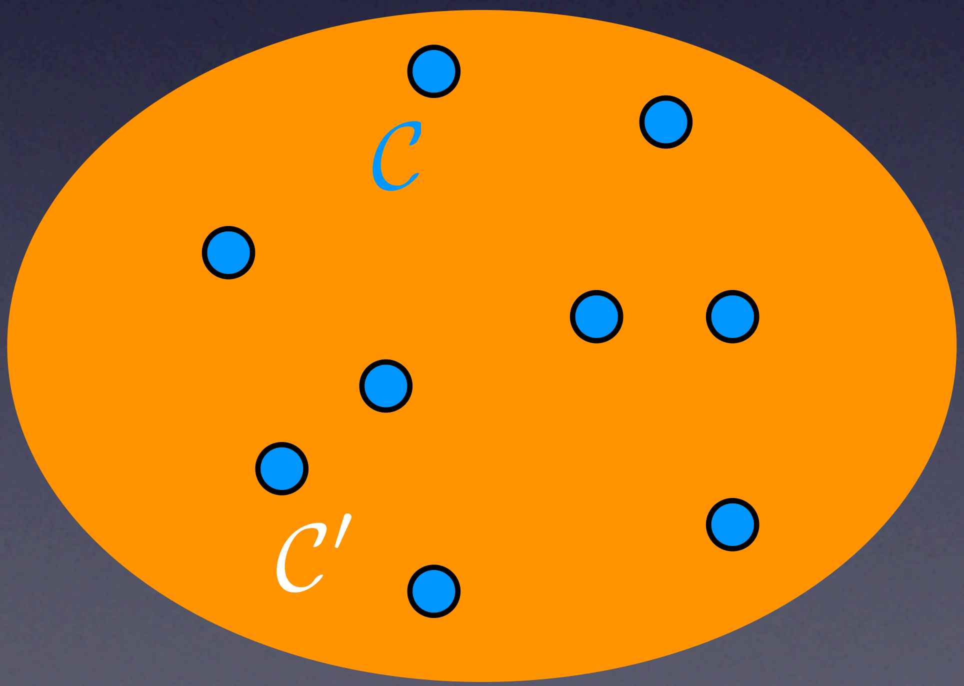
$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \leq 0 \quad 1 \leq i \leq m, \\ & h_i(\mathbf{x}) = 0 \quad 1 \leq i \leq p \end{aligned}$$

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

Dual

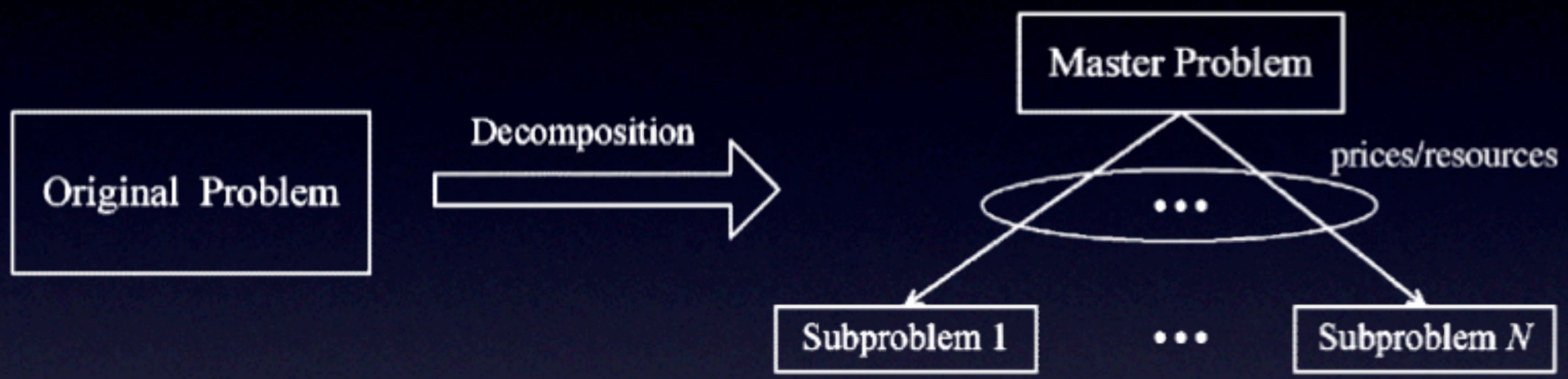
$$\begin{aligned} \underset{\boldsymbol{\lambda}, \boldsymbol{\nu}}{\text{maximize}} \quad & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned}$$

Convex Relaxations



Optimization Theory and Distributed Solutions

Decomposition framework



$$\begin{array}{ll} \text{minimize} & f_1(x_1) + f_2(x_2) \\ \text{subject to} & x_1 \in \mathcal{C}_1, \quad x_2 \in \mathcal{C}_2 \\ & h_1(x_1) + h_2(x_2) \preceq 0 \end{array}$$

Primal decomposition (resources)

$$\begin{array}{ll} \text{minimize} & f_1(x_1) \\ \text{subject to} & x_1 \in \mathcal{C}_1, \quad h_1(x_1) \preceq t, \end{array}$$

$$\begin{array}{ll} \text{minimize} & f_2(x_2) \\ \text{subject to} & x_2 \in \mathcal{C}_2, \quad h_2(x_2) \preceq -t. \end{array}$$

Dual decomposition (prices)

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + \lambda^T h_1(x_1) \\ \text{subject to} & x_1 \in \mathcal{C}_1, \end{array}$$

$$\begin{array}{ll} \text{minimize} & f_2(x_2) + \lambda^T h_2(x_2) \\ \text{subject to} & x_2 \in \mathcal{C}_2. \end{array}$$

Sorting as a Mathematical Program

Given a list of numbers

$$\{u_1, u_2, \ldots, u_N\}$$

find a permutation such that

$$u_{\pi(1)} \leq u_{\pi(2)} \leq \ldots \leq u_{\pi(N)}$$

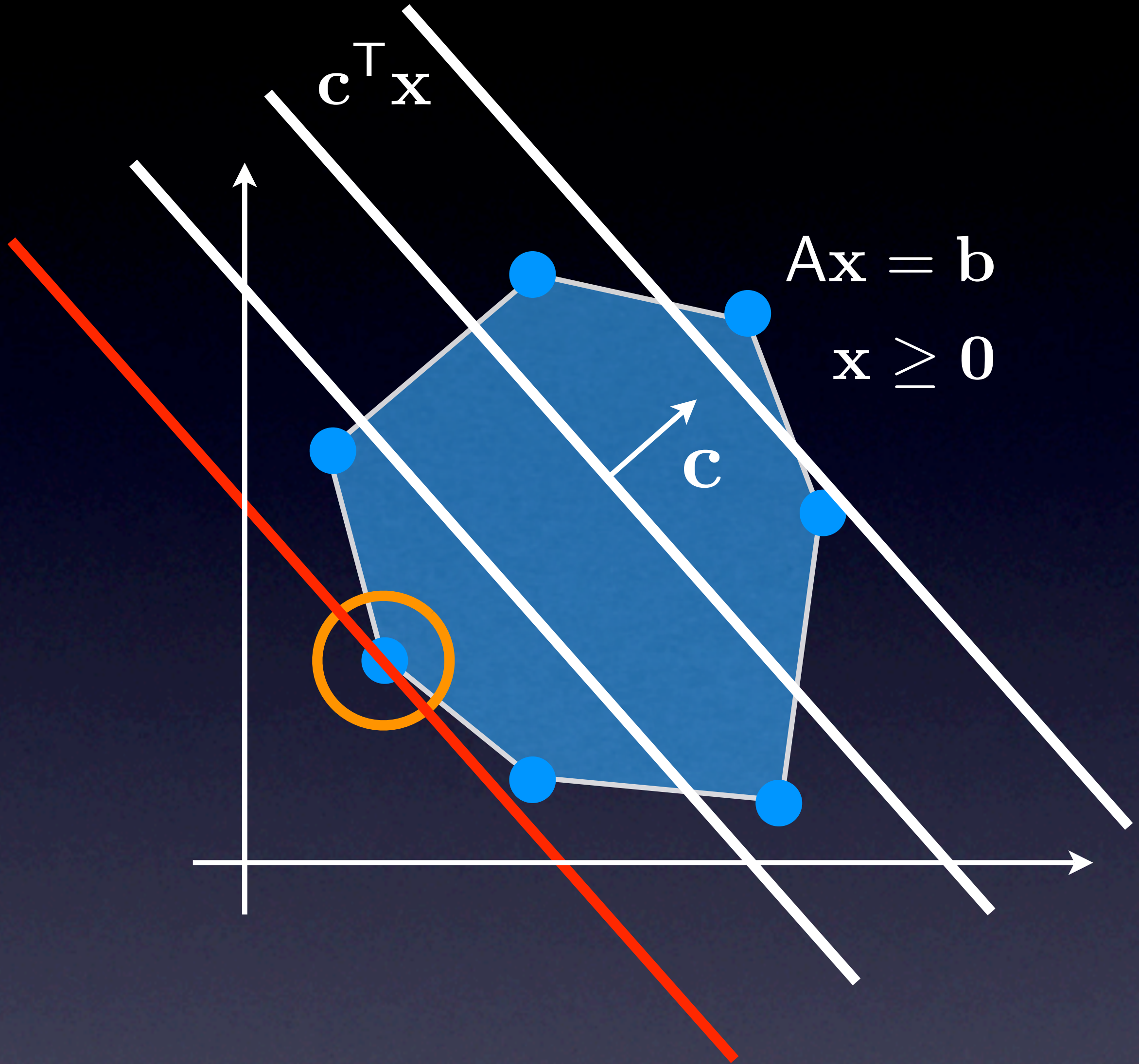
Optimization problem

$$f_{\mathbf{u}} : \mathcal{S}_N \longrightarrow \mathbb{R}$$

$$\pi \longmapsto f_{\mathbf{u}}(\pi)$$

$f_{\mathbf{u}}$ reaches the maximum when the permutation sorts the list

Sorting as a Mathematical Program

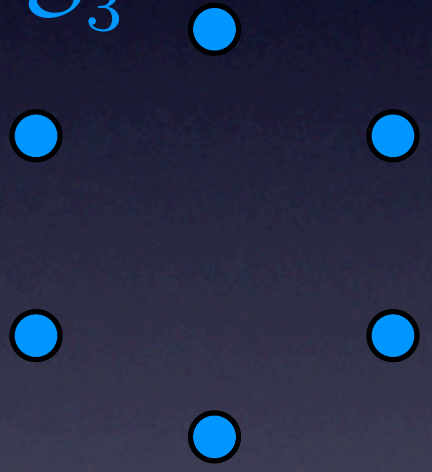


Permutation matrices

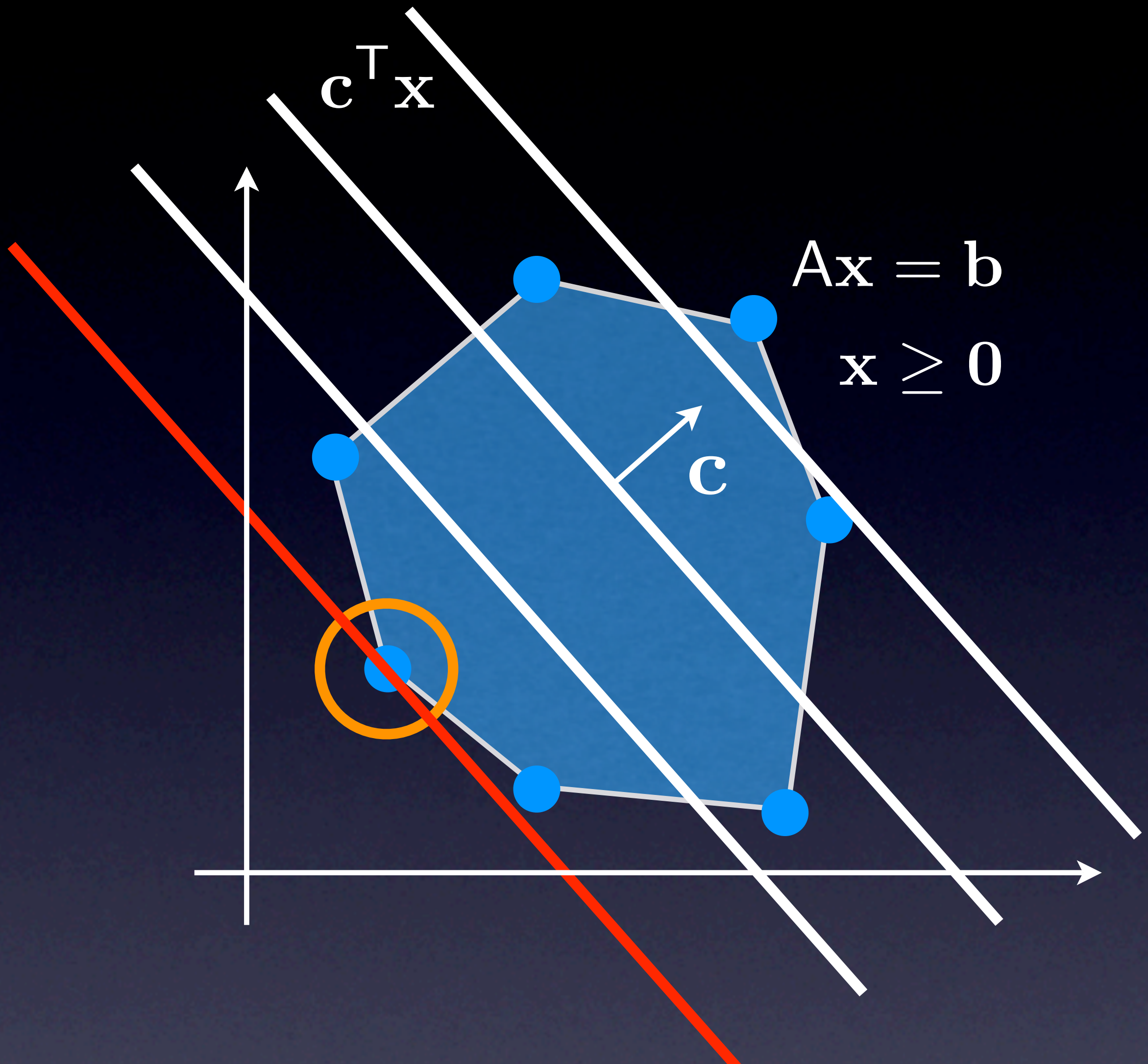
$$\sum_{i=1}^{N} \mathsf{P}_{ij} = \sum_{j=1}^{N} \mathsf{P}_{ij} = 1$$

$$\mathsf{P}_{ij} \in \{0, 1\}$$

$\mathcal{S}_3$



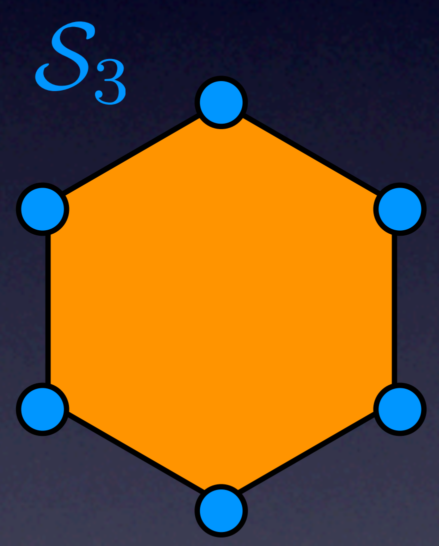
Sorting as a Mathematical Program



Permutation matrices

$$\sum_{i=1}^{N} \mathsf{P}_{ij} = \sum_{j=1}^{N} \mathsf{P}_{ij} = 1$$

$$\mathsf{P}_{ij} \geq 0$$



<u>Birkhoff-von Neumann Theorem</u>: The convex hull of the set of permutation matrices is the set of doubly stochastic matrices

Sorting as a Mathematical Program

Linear program $f_{\mathbf{u}}(\pi) = 1u_{\pi(1)} + 2u_{\pi(2)} + \cdots + Nu_{\pi(N)}$

maximize $\sum_{i=1}^{N} i \sum_{j=1}^{N} \mathsf{P}_{ij} u_j$

subject to $\sum_{j=1}^{N} \mathsf{P}_{ij} = 1,\ i = 1, \ldots, N,$

$\sum_{i=1}^{N} \mathsf{P}_{ij} = 1,\ j = 1, \ldots, N,$

$\mathsf{P}_{ij} \geq 0,\ i, j = 1, \ldots, N.$ Convex relaxation

All the structure of the permutation is encoded on the constraints

DUALITY: Relax the original problem by transfering the constraints to the objective function

Sorting as a Mathematical Program

Dual linear program

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} r_i + \sum_{j=1}^{N} c_j \\ \text{subject to} & r_i + c_j \geq i u_j \quad i, j = 1, \ldots, N. \end{array}$$

Auction algorithm

Bidding phase:

$$\text{Best value} \rightarrow v_{ij^*} = \max_j (i u_j - c_j),$$

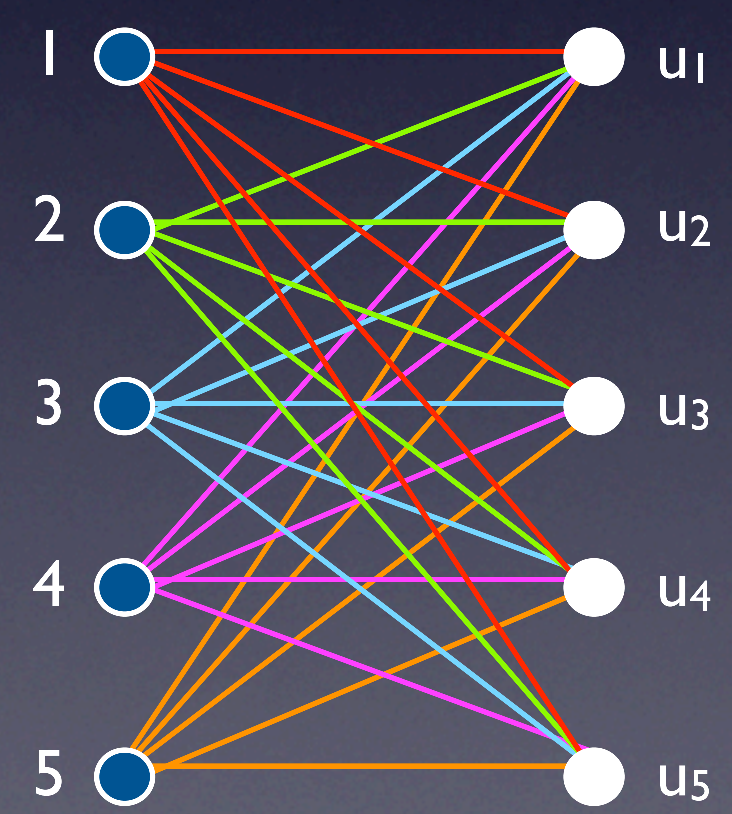
$$\text{Second best value} \rightarrow w_{ij^*} = \max_{j \neq j^*} (i u_j - c_j),$$

$$\text{Bid} \rightarrow b_i = p_i + v_{ij^*} - w_{ij^*} + \epsilon$$

Assignment phase:

$$c_j = \max_i b_{ij},$$

$$i_j = \arg\max_i b_{ij} \Rightarrow \mathsf{P}_{ij} := \begin{cases} 1, & \text{if } i = i_j, \\ 0, & \text{otherwise.} \end{cases}$$



Sorting as a Mathematical Program

Another approach

(Permutation as a product of standard transpositions)

$$\begin{array}{ll} \text{maximize} & f_{\mathbf{u}}(\pi) = [1, 2, \ldots, N] \mathsf{E}_{N/2} \mathsf{O}_{N/2} \cdots \mathsf{E}_1 \mathsf{O}_1 \mathbf{u} \\ \text{subject to} & \boxed{0 \le c_{kl} \le 1, \ k = 1, \ldots, N/2, \ l = 1, \ldots, N.} \end{array}$$

Convex Relaxation

Gauss-Seidel iteration

$$\mathsf{O}_k^{(t+1)} = \arg\min_{\mathsf{O}} [1, 2, \ldots, N] \underbrace{\mathsf{E}_{N/2}^{(t)} \mathsf{O}_{N/2}^{(t)} \cdots \mathsf{E}_{k+1}^{(t)} \mathsf{O}_{k+1}^{(t)} \mathsf{E}_k^{(t)}}_{(t)} \, \mathsf{O} \, \underbrace{\mathsf{E}_{k-1}^{(t+1)} \mathsf{O}_{k-1}^{(t+1)} \cdots \mathsf{E}_1^{(t+1)} \mathsf{O}_1^{(t+1)}}_{(t+1)} \mathbf{u}$$

$$\mathsf{E}_k^{(t+1)} = \arg\min_{\mathsf{E}} [1, 2, \ldots, N] \underbrace{\mathsf{E}_{N/2}^{(t)} \mathsf{O}_{N/2}^{(t)} \cdots \mathsf{E}_{k+1}^{(t)} \mathsf{O}_{k+1}^{(t)}}_{(t)} \, \mathsf{E} \, \underbrace{\mathsf{O}_k^{(t+1)} \mathsf{E}_{k-1}^{(t+1)} \mathsf{O}_{k-1}^{(t+1)} \cdots \mathsf{E}_1^{(t+1)} \mathsf{O}_1^{(t+1)}}_{(t+1)} \mathbf{u}$$

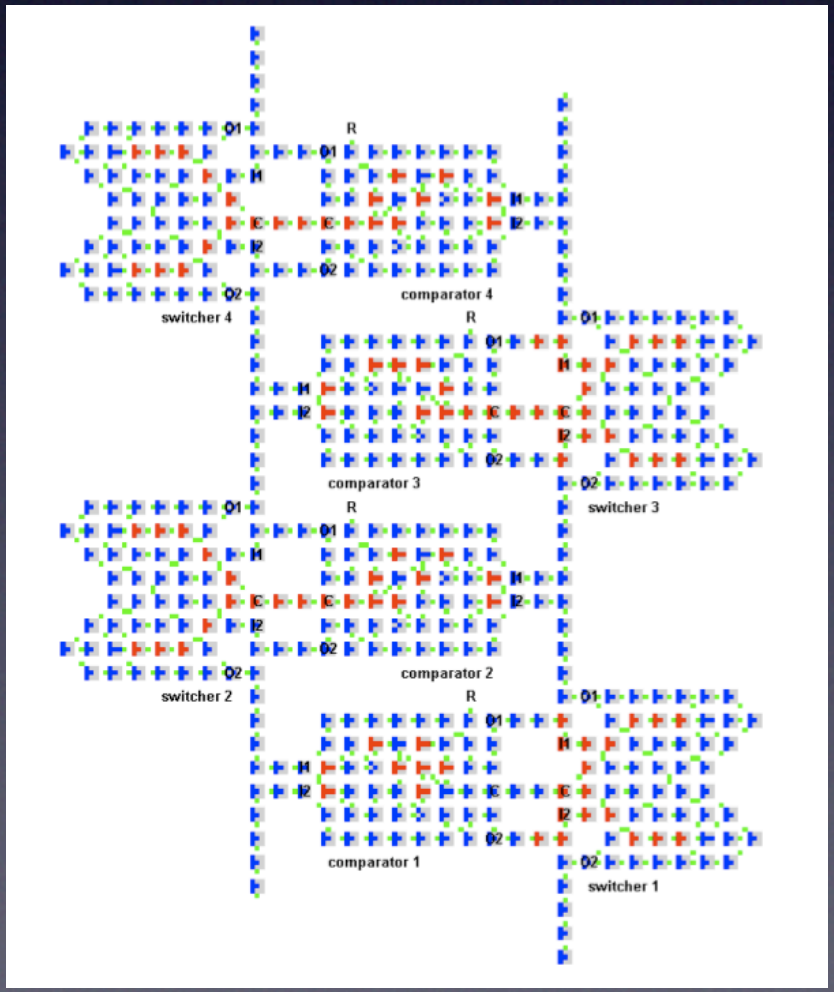
Sorting as a Mathematical Program

Another approach

(Permutation as a product of standard transpositions)

$$\begin{array}{ll} \text{maximize} & f_{\mathbf{u}}(\pi) = [1, 2, \ldots, N]\mathsf{E}_{N/2}\mathsf{O}_{N/2} \cdots \mathsf{E}_1\mathsf{O}_1\mathbf{u} \\ \text{subject to} & 0 \leq c_{kl} \leq 1, \; k = 1, \ldots, N/2, \; l = 1, \ldots, N. \end{array}$$

Convex Relaxation



Sudoku as a Linear Program

$$x_{ijk} = \begin{cases} 1 & \text{if symbol in cell } (i,j) \text{ is } s_k, \\ 0 & \text{otherwise.} \end{cases}$$

Blank Sudoku

cells: $$\sum_{k=1}^{n} x_{ijk} = 1, \qquad \text{for all } i, j = 1, \ldots, n.$$

rows: $$\sum_{j=1}^{n} x_{ijk} = 1, \qquad \text{for all } i, k = 1, \ldots, n.$$

cols: $$\sum_{i=1}^{n} x_{ijk} = 1, \qquad \text{for all } j, k = 1, \ldots, n.$$

blocks: $$\sum_{i=(I-1)m+1}^{Im} \sum_{j=(J-1)m+1}^{Jm} x_{ijk} = 1, \qquad \text{for all } I, J = 1, \ldots, m \quad k = 1, \ldots, n.$$

2			4		9		7	5
						8	2	
	8		1		5	9		3
9	2	7		4		6		8
8		3		9		5	4	1
5		2	3		6		8	
	9	8						
6	4		9		8			2

$$x_{11k} = \delta_{2k},\ x_{14k} = \delta_{4k},\ x_{16k} = \delta_{9k}, \ldots$$

Sudoku as a Linear Program

Formulation 1:

Givens:

$$x_{pqr} = 0, \quad \text{if any of these} \begin{cases} p = i, q = j, r \neq k & \text{(cell)}, \\ p \neq i, q = j, r = k & \text{(row)}, \\ p = i, q \neq j, r = k & \text{(column)}, \\ \lfloor p/m \rfloor = \lfloor i/m \rfloor, \lfloor q/m \rfloor = \lfloor j/m \rfloor, r = k & \text{(block)}, \end{cases}$$

$$x_{pqr} = 1, \quad \text{if } p = i, q = j, r = k.$$

$$\begin{array}{ll} \text{find} & \mathbf{x} \\ \text{subject to} & \mathbf{x} \in \mathcal{B} \cap \mathcal{G}, \\ & \mathbf{x} \in \{0,1\}^{n^2}, \end{array}$$

Sudoku as a Linear Program

Formulation 1:

Givens:

$$x_{pqr} = 0, \quad \text{if any of these} \begin{cases} p = i, q = j, r \neq k & \text{(cell)}, \\ p \neq i, q = j, r = k & \text{(row)}, \\ p = i, q \neq j, r = k & \text{(column)}, \\ \lfloor p/m \rfloor = \lfloor i/m \rfloor, \lfloor q/m \rfloor = \lfloor j/m \rfloor, r = k & \text{(block)}, \end{cases}$$

$$x_{pqr} = 1, \quad \text{if } p = i, q = j, r = k.$$

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \equiv 0 \\ \text{subject to} & \mathbf{x} \in \mathcal{B} \cap \mathcal{G}, \\ & \mathbf{x} \in \{0, 1\}^{n^2}. \end{array}$$

Sudoku as a Linear Program

Formulation 1:

Givens:

$$x_{pqr} = 0, \quad \text{if any of these} \begin{cases} p = i, q = j, r \neq k & \text{(cell)}, \\ p \neq i, q = j, r = k & \text{(row)}, \\ p = i, q \neq j, r = k & \text{(column)}, \\ \lfloor p/m \rfloor = \lfloor i/m \rfloor, \lfloor q/m \rfloor = \lfloor j/m \rfloor, r = k & \text{(block)}, \end{cases}$$

$$x_{pqr} = 1, \quad \text{if } p = i, q = j, r = k.$$

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \equiv 0 \\ \text{subject to} & \mathbf{x} \in \mathcal{B} \cap \mathcal{G}, \\ & 0 \leq \mathbf{x}, \end{array}$$

Sudoku as a Linear Program

Formulation 2:

$$\begin{array}{ll} \text{minimize} & f_G(\mathbf{x}) \\ \text{subject to} & \mathbf{x} \in \mathcal{B}, \\ & \mathbf{x} \in \{0,1\}^{n^2}. \end{array}$$

$$f_G(\mathbf{x}) \equiv \sum_{i,j,k=1}^{n} c_{ijk} x_{ijk}$$

$$c_{ijk} = \begin{cases} -1 & \text{if there is a given } k \text{ in cell } (i,j), \\ 1 & \text{if there is a given } \neq k \text{ at } (i,j) \text{ or } = k \text{ at } (i, \neq j) \text{ or at } (\neq i, j), \\ 0 & \text{otherwise} \end{cases}$$

Sudoku as a Linear Program

Formulation 2:

$$
\begin{array}{ll}
\text{minimize} & f_G(\mathbf{x}) \\
\text{subject to} & \mathbf{x} \in \mathcal{B}, \\
 & 0 \leq \mathbf{x},
\end{array}
$$

$$f_G(\mathbf{x}) \equiv \sum_{i,j,k=1}^{n} c_{ijk} x_{ijk}$$

$$c_{ijk} = \begin{cases} -1 & \text{if there is a given } k \text{ in cell } (i,j), \\ 1 & \text{if there is a given } \neq k \text{ at } (i,j) \text{ or } = k \text{ at } (i, \neq j) \text{ or at } (\neq i, j), \\ 0 & \text{otherwise} \end{cases}$$

Sudoku as a Linear Program

Algorithms:

GLPK (GNU Linear Programming Kit)

http://www.gnu.org/software/glpk/

1. Simplex (several flavours)

2. Interior point (primal-dual)

Transformations:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathrm{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \quad \Rightarrow \quad \begin{aligned} \text{minimize} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathrm{AP}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Sudoku as a Lattice Gas

Multicomponent non-additive hard-core lattice gas

2			4		9		7	5
						8	2	
	8		1		5	9		3
9	2	7		4		6		8
8		3		9		5	4	1
5		2	3		6		8	
	9	8						
6	4		9		8			2

Sudoku as a Lattice Gas

Free-energy density Functional Theory

$$\mathcal{F}[\rho_{ijk}] = \sum \{\phi(\mathrm{ROW}_{ik}) + \phi(\mathrm{COL}_{jk}) + \phi(\mathrm{BLOCK}_{IJ}) + \phi(\mathrm{POINT}_{ij}) \\ -\phi(\mathrm{row}_{ik:IJ}) - \phi(\mathrm{col}_{jk:IJ}) - \phi(\mathrm{point}_{ijk})\}$$

$$\phi(\eta) = \eta \ln \eta + (1-\eta)\ln(1-\eta)$$

Euler-Lagrange Eqs.:

$$\rho_{ijk}(1-\rho_{ijk})(1-\sum_{l:B(i,j)}\rho_{ilk})(1-\sum_{l:B(i,j)}\rho_{ljk}) = z_{ijk}(1-\sum_{l}\rho_{ijl})(1-\sum_{l}\rho_{ilk}) \\ \times(1-\sum_{l}\rho_{ljk})(1-\sum_{l,m:B(i,j)}\rho_{lmk})$$

Sudoku as a Quadratic Program

$$\begin{array}{ll} \text{maximize} & \|\mathbf{x}\|_2^2 \\ \text{subject to} & \mathbf{x} \in \mathcal{B} \cap \mathcal{G} \\ & \mathbf{x} \geq \mathbf{0} \end{array}$$

Non-convex problem!

NP-hard if the quadratic term matrix is definite negative

Some Final Comments

- LP vs. Semidefinite Relaxations.
- Distributed Solvers.
- Optimization methods as a tool to derive local Message Passing algortihms.
- Connections with BP-like algorithms