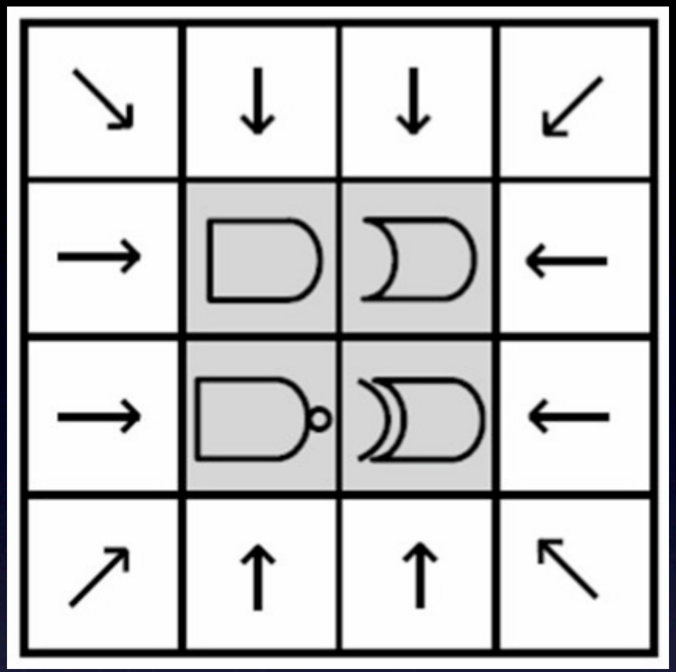
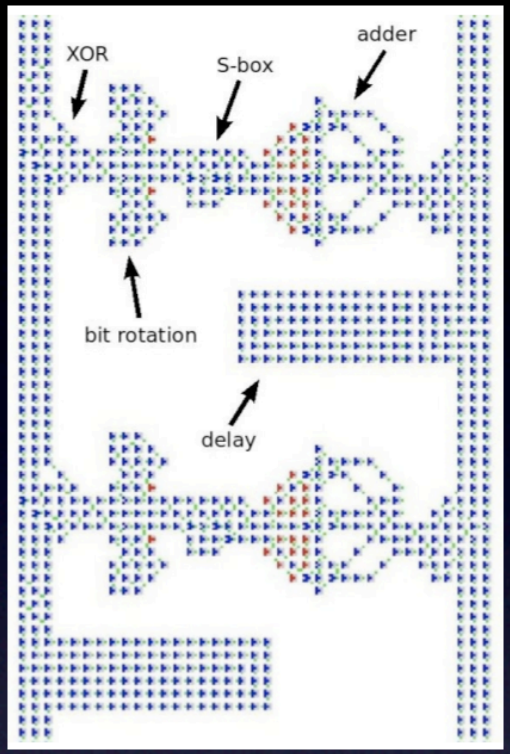
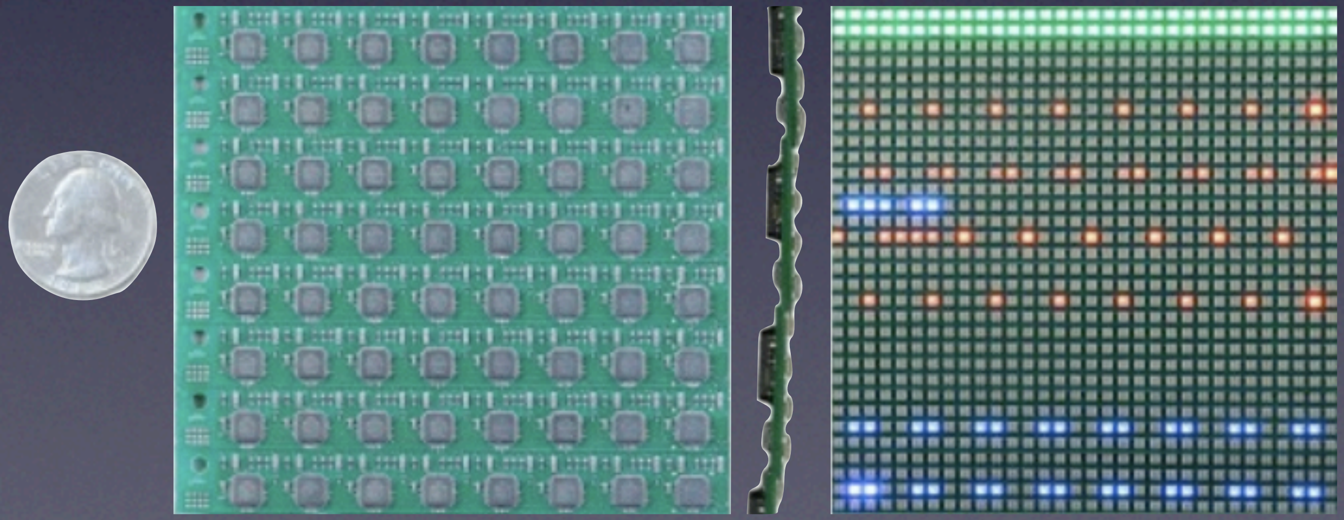
Mathematical Programming

Luis Lafuente
Center for Bits and Atoms
Massachusetts Institute of Technology

Cellular Microcode







Hardware

How to program a conformal computer?

Symmetries in Physics ➡ Global ⇔ Local

- Global ↙ Variational principles
- Local ↘ Euler-Lagrange Equations

Symmetries in Conformal Computing ➡ Global ⇔ Local

How to program a conformal computer?

Constraint Programming

Programs as a set of constraints between variables

Not good solvers

How to program a conformal computer?

Constraint Programming

Programs as a set of constraints between variables

Not good solvers

Mathematical Programming

Same philosophy, but GOOD solvers

- Lagrange Duality
- Convex relaxations
- Decomposition methods
- ...

→ Scalable

Distributed

How to program a conformal computer?

Constraint Programming

Programs as a set of constraints between variables

Not good solvers

Mathematical Programming

Local rules as distributed solutions of constrained global optimization problems

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

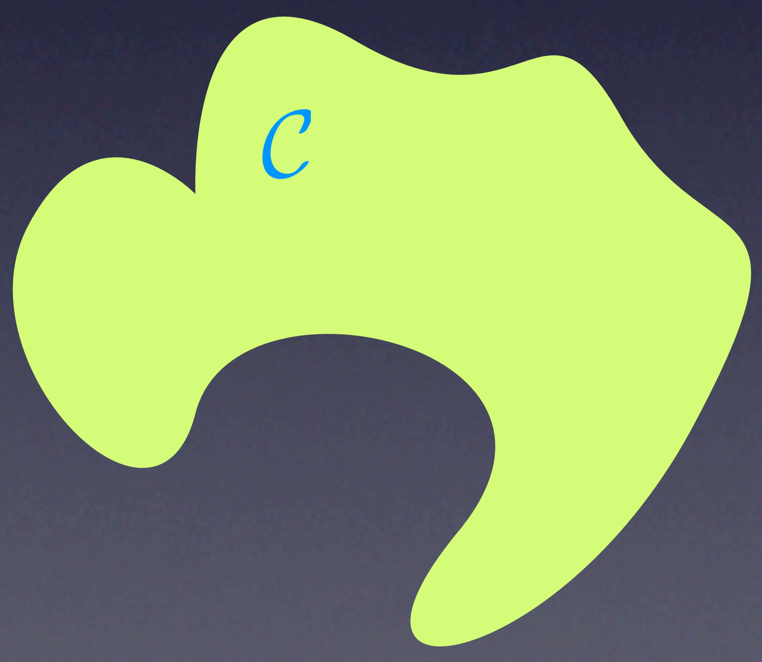
$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) \\ \text{subject to} & f_i(\mathbf{x}) \leq 0 \quad 1 \leq i \leq m, \\ & h_i(\mathbf{x}) = 0 \quad 1 \leq i \leq p \end{array}$$

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

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

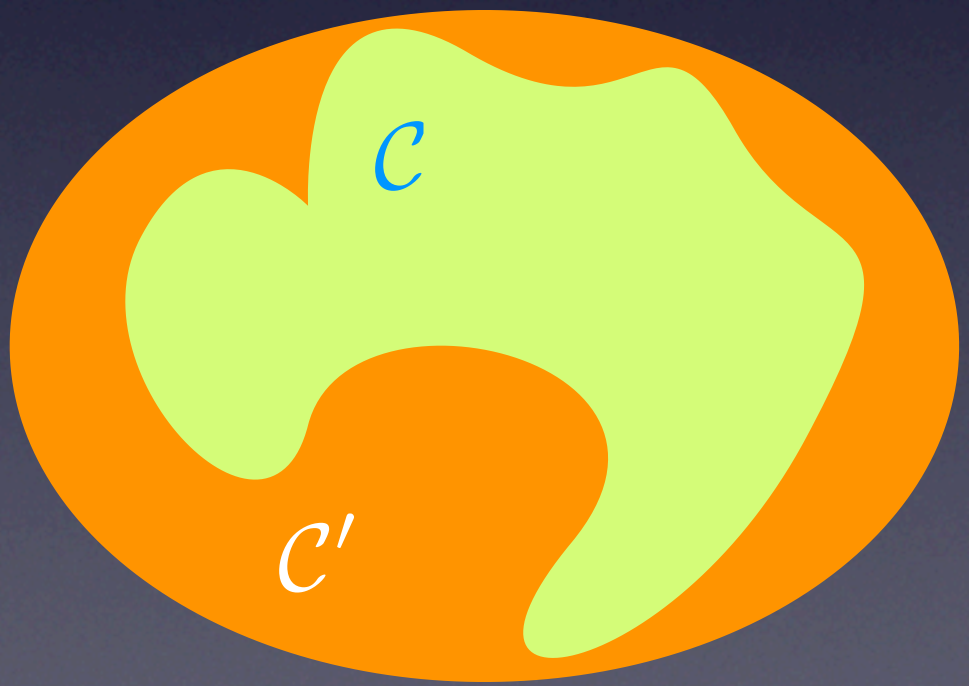
$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_i(\mathbf{x}) \le 0 \quad 1 \le i \le m, \\ & h_i(\mathbf{x}) = 0 \quad 1 \le i \le p \end{aligned}$$

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

Dual

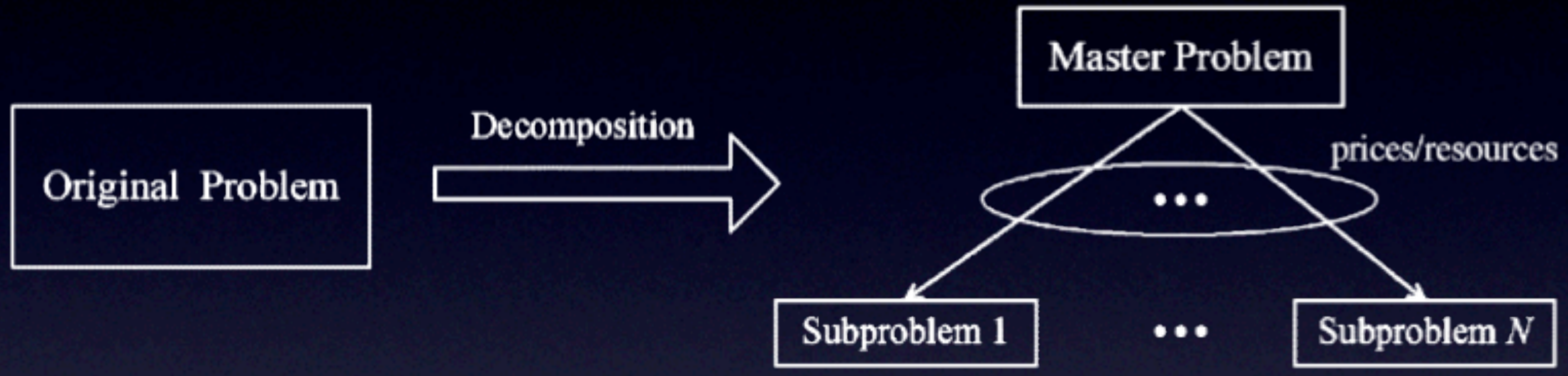
$$\begin{aligned} \underset{\boldsymbol{\lambda}, \boldsymbol{\nu}}{\text{maximize}} \quad & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \ge 0 \end{aligned}$$

Convex Relaxations



Optimization Theory and Convexity

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)

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

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

High Level Language: Mathematical Programming

Formulation

↓

Preprocessing ➜ Objective function and constraints

↓

Method ➜ Primal, dual, primal-dual, simplex, interior point, conjugated gradients, decomposition methods,...

↓

Local Rules

↓

Compiled program

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

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:

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

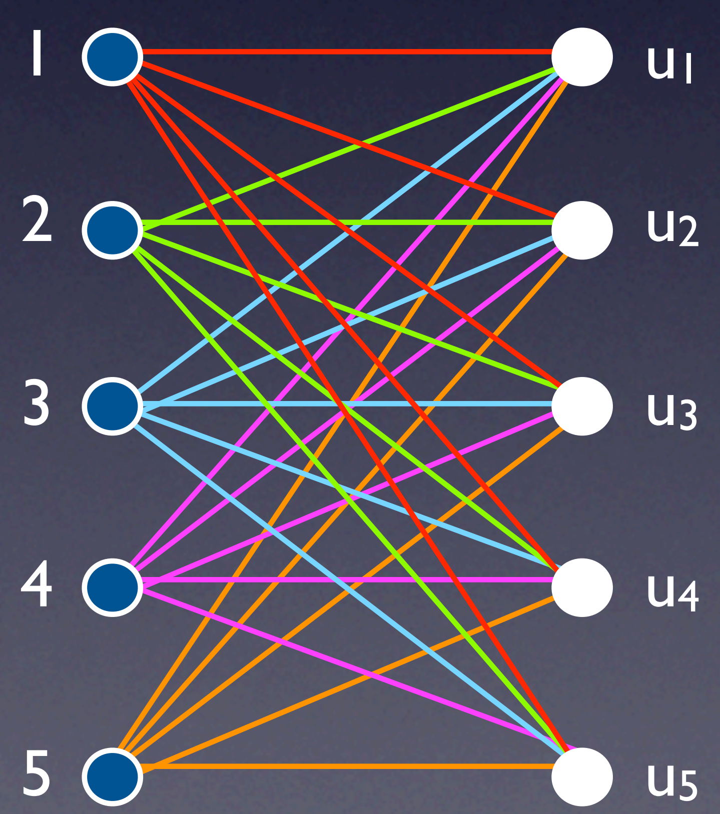
$$\text{Second best value} \rightarrow w_{ij^*} = \max_{j \neq j^*} (i u_j - c_j),$$

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

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)

Theorem:

Every permutation $\pi \in \mathcal{S}_N$ can be written as the product

$$\pi = \prod_{k=1}^{N/2} (1 \;\; 1 + c_{k1})(3 \;\; 3 + c_{k3}) \cdots (N-1 \;\; N-1 + c_{k\,N-1})$$

$$(2 \;\; 2 + c_{k2})(4 \;\; 4 + c_{k4}) \cdots (N-1 \;\; N-2 + c_{k,N-2}),$$

where the c_{ij}'s are 0 or 1, and $(i \;\; i) = 12 \ldots N$

Example

$3412 {=} (2 \;\; 3)(1 \;\; 2)(3 \;\; 4)(2 \;\; 3) \;\; \Rightarrow \;\; c_{11} = c_{31} = 0, \;\; c_{12} = c_{21} = c_{23} = c_{22} = 1.$

Sorting as a Mathematical Program

Another approach

(Permutation as a product of standard transpositions)

$$f_{\mathbf{u}}(\pi) = [1, 2, \ldots, N] \underbrace{\mathsf{E}_{N/2}\mathsf{O}_{N/2} \cdots \mathsf{E}_1\mathsf{O}_1}_{\mathsf{P}} \mathbf{u}$$

$$\mathsf{O}_k = \begin{bmatrix} \mathsf{C}_{k1} & & & \\ & \mathsf{C}_{k3} & & \\ & & \ddots & \\ & & & \mathsf{C}_{k\,N-1} \end{bmatrix} \quad \mathsf{E}_k = \begin{bmatrix} 1 & & & & \\ & \mathsf{C}_{k2} & & & \\ & & \ddots & & \\ & & & \mathsf{C}_{k\,N-2} & \\ & & & & 1 \end{bmatrix}$$

$$\mathsf{C}_{kl} = \begin{bmatrix} 1 - c_{kl} & c_{kl} \\ c_{kl} & 1 - c_{kl} \end{bmatrix}$$

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

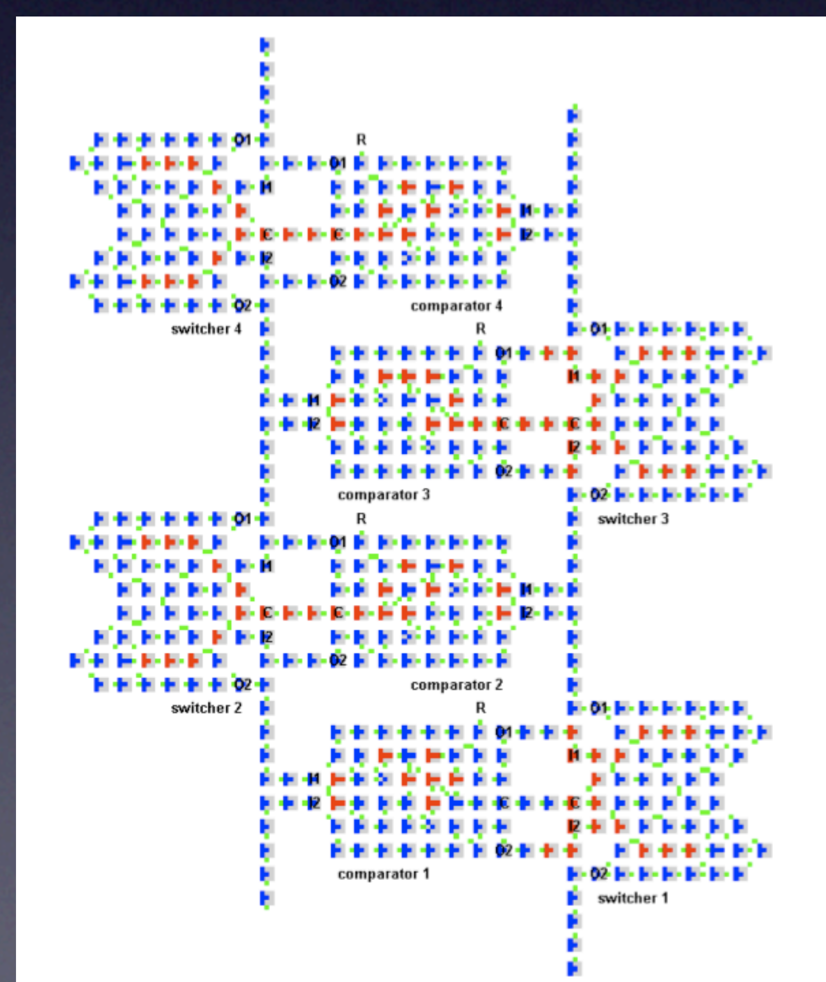
Another approach

(Permutation as a product of standard transpositions)

$$\text{maximize} \quad f_{\mathbf{u}}(\pi) = [1, 2, \ldots, N]\mathsf{E}_{N/2}\mathsf{O}_{N/2} \cdots \mathsf{E}_1\mathsf{O}_1\mathbf{u}$$

$$\text{subject to} \quad 0 \leq c_{kl} \leq 1,\ k = 1, \ldots, N/2,\ l = 1, \ldots, N.$$

Convex Relaxation



General Optimization Solvers in the Boolean CA

Building blocks: Linear Algebra

Matrix multiplication, Cholesky, LU, QR, SVD,...

ScaLAPACK (Scalable LAPACK)

PLASMA (Parallel LA for Scalable Multi-core Architectures)

General Optimization Solvers in the Boolean CA

Building blocks: Linear Algebra

Matrix multiplication, Cholesky, LU, QR, SVD,...

ScaLAPACK (Scalable LAPACK)
PLASMA (Parallel LA for Scalable Multi-core Architectures)

↓

Parallel at the PROCESSOR level

General Optimization Solvers in the Boolean CA

Building blocks: Linear Algebra

Matrix multiplication, Cholesky, LU, QR, SVD,...

ScaLAPACK (Scalable LAPACK)

PLASMA (Parallel LA for Scalable Multi-core Architectures)

Parallel at the PROCESSOR level

Parallel Linear Algebra in a Conformal Computer

↓

Parallel at the BIT level

General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$A\mathbf{x} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$A \times = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

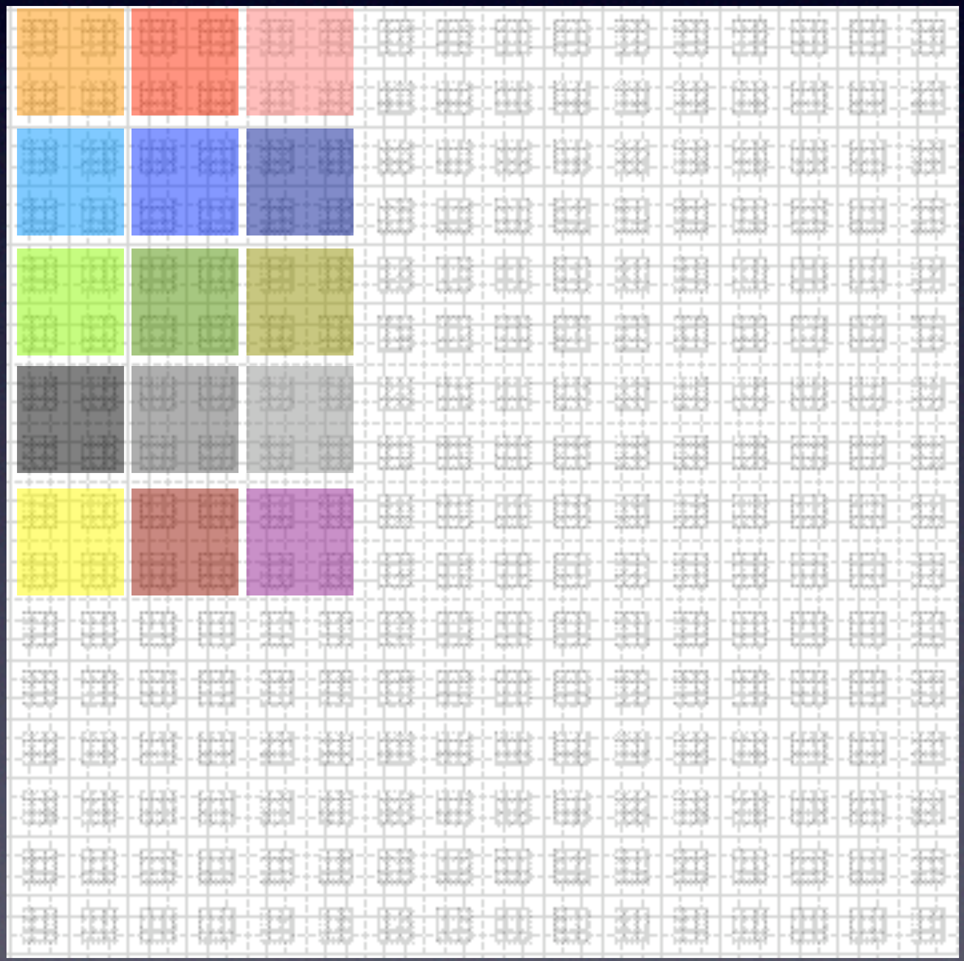
General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$\mathrm{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\mathrm{x} \quad \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$



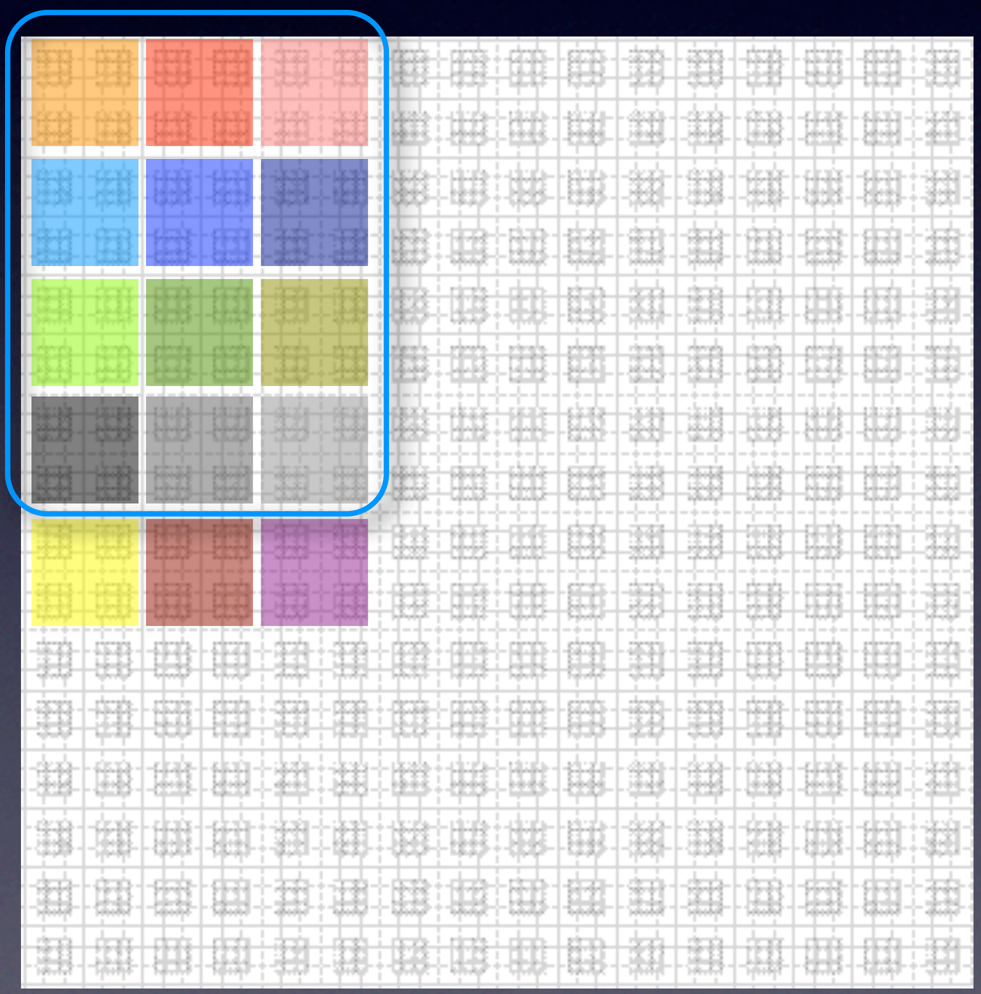
General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

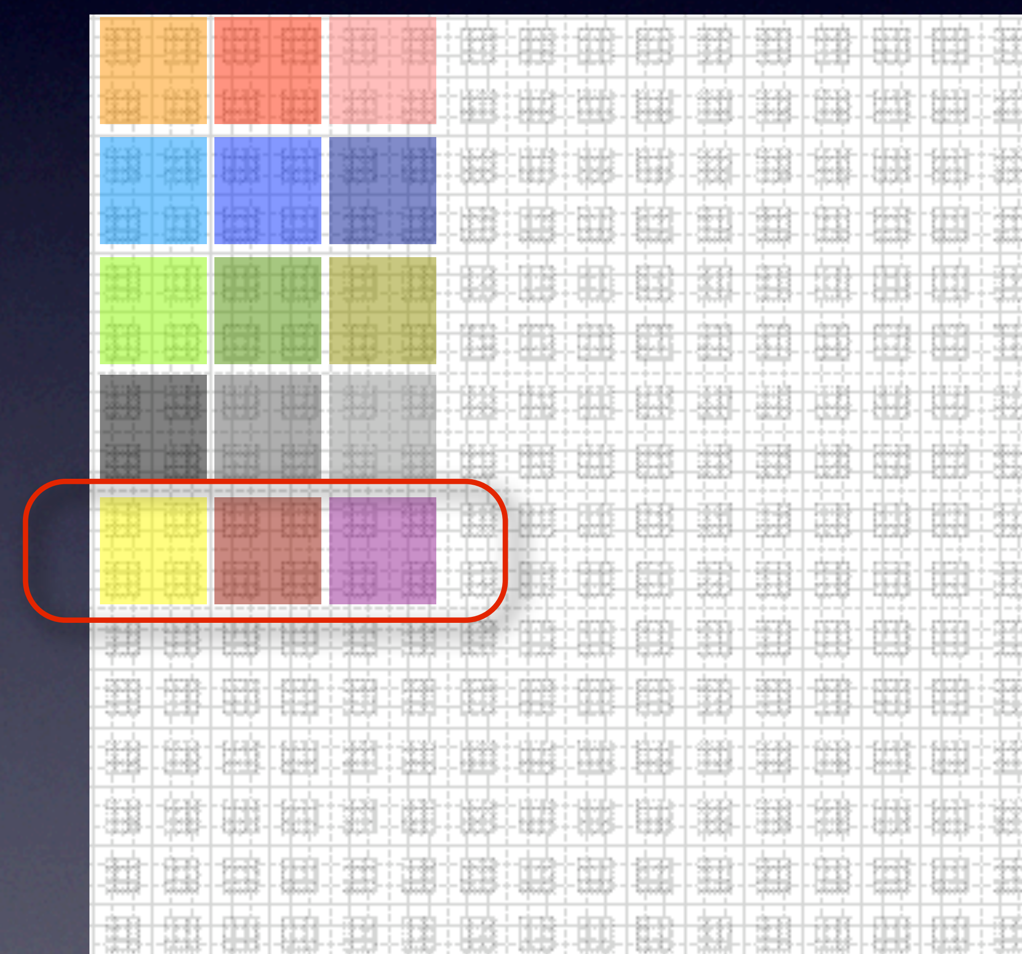


General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A \times = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$



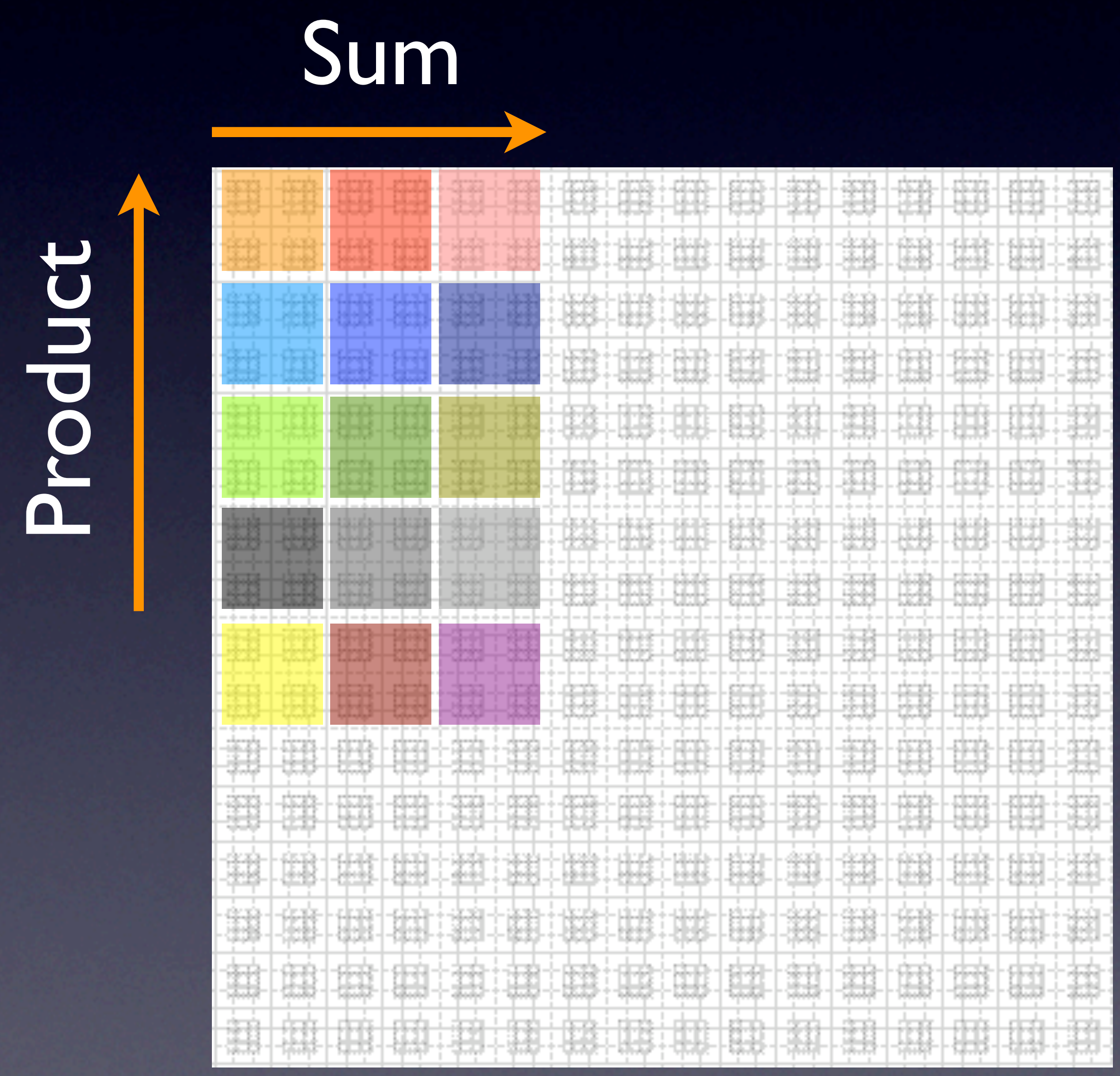
General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$A\,x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

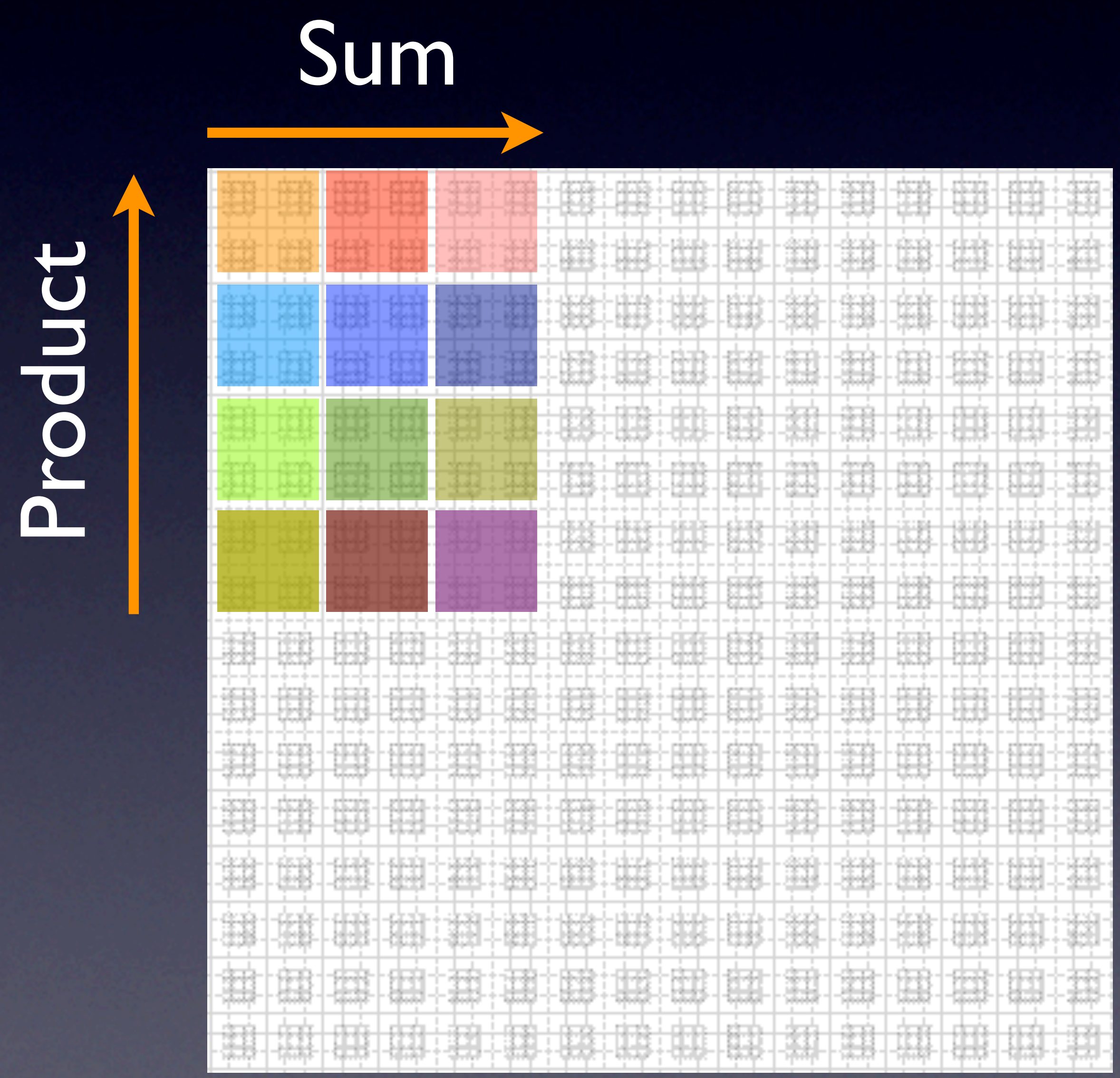


General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$A \times = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$



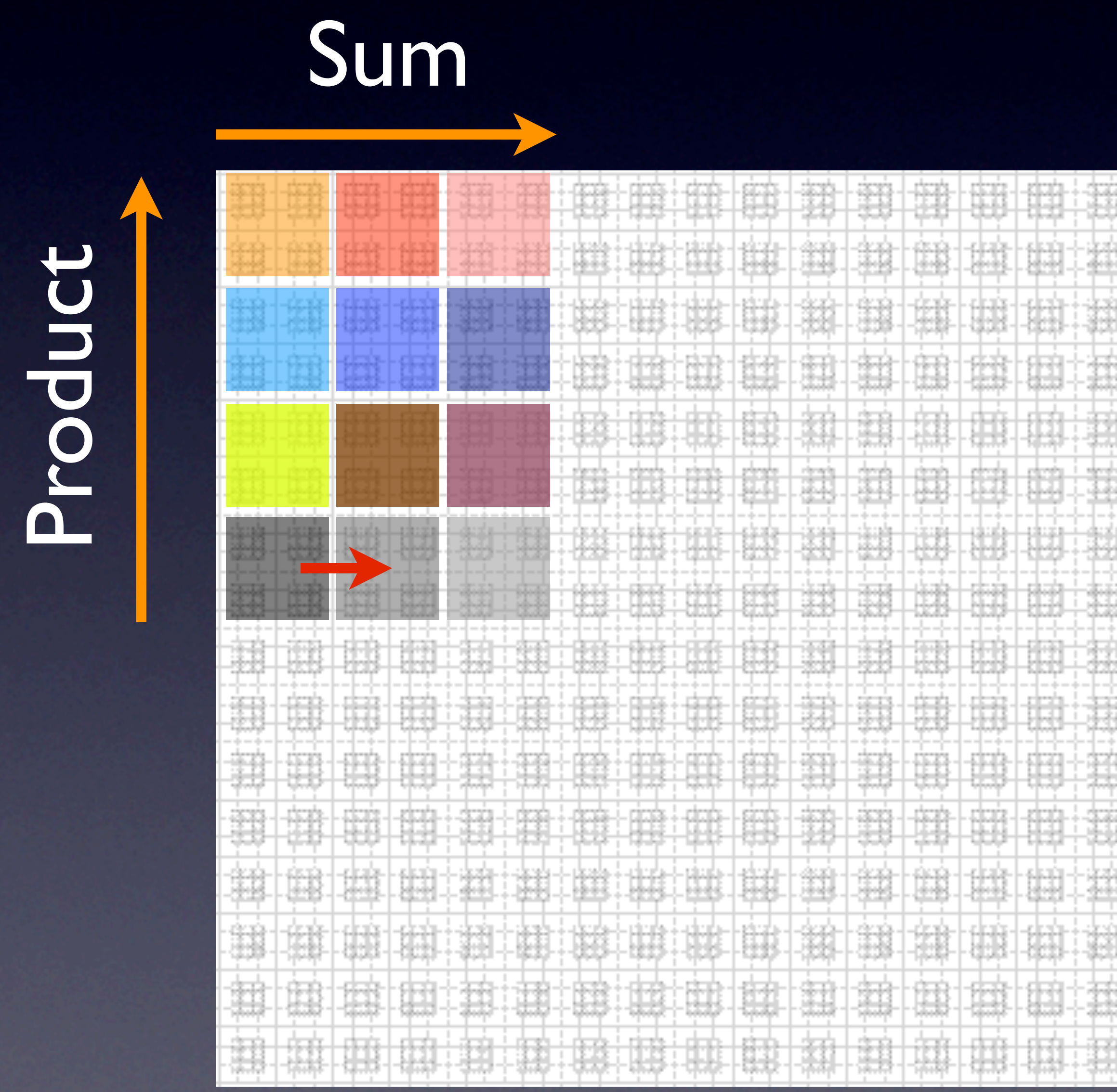
General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A \; x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

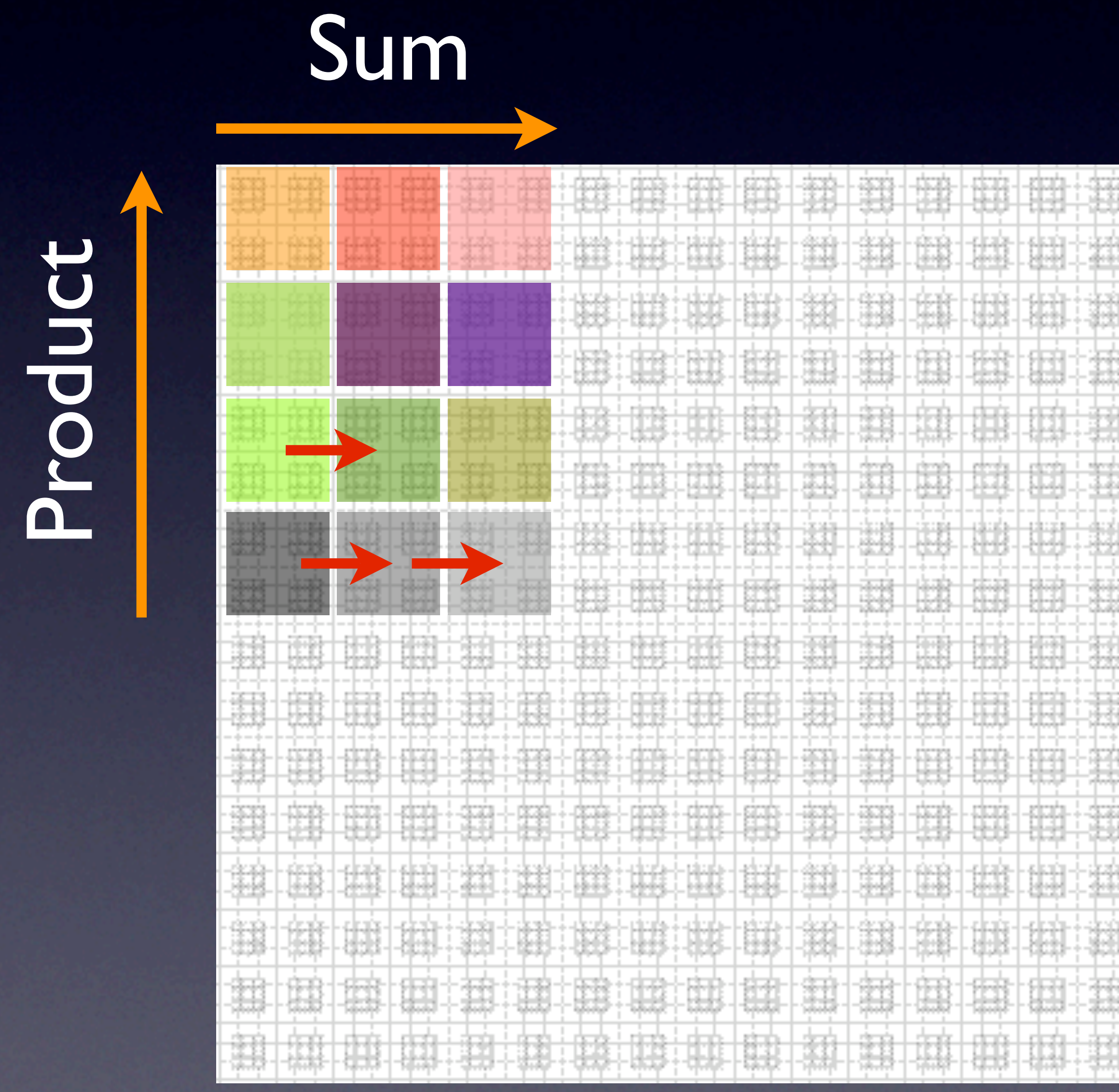


General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$\begin{matrix} A \\ x \end{matrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \quad \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$



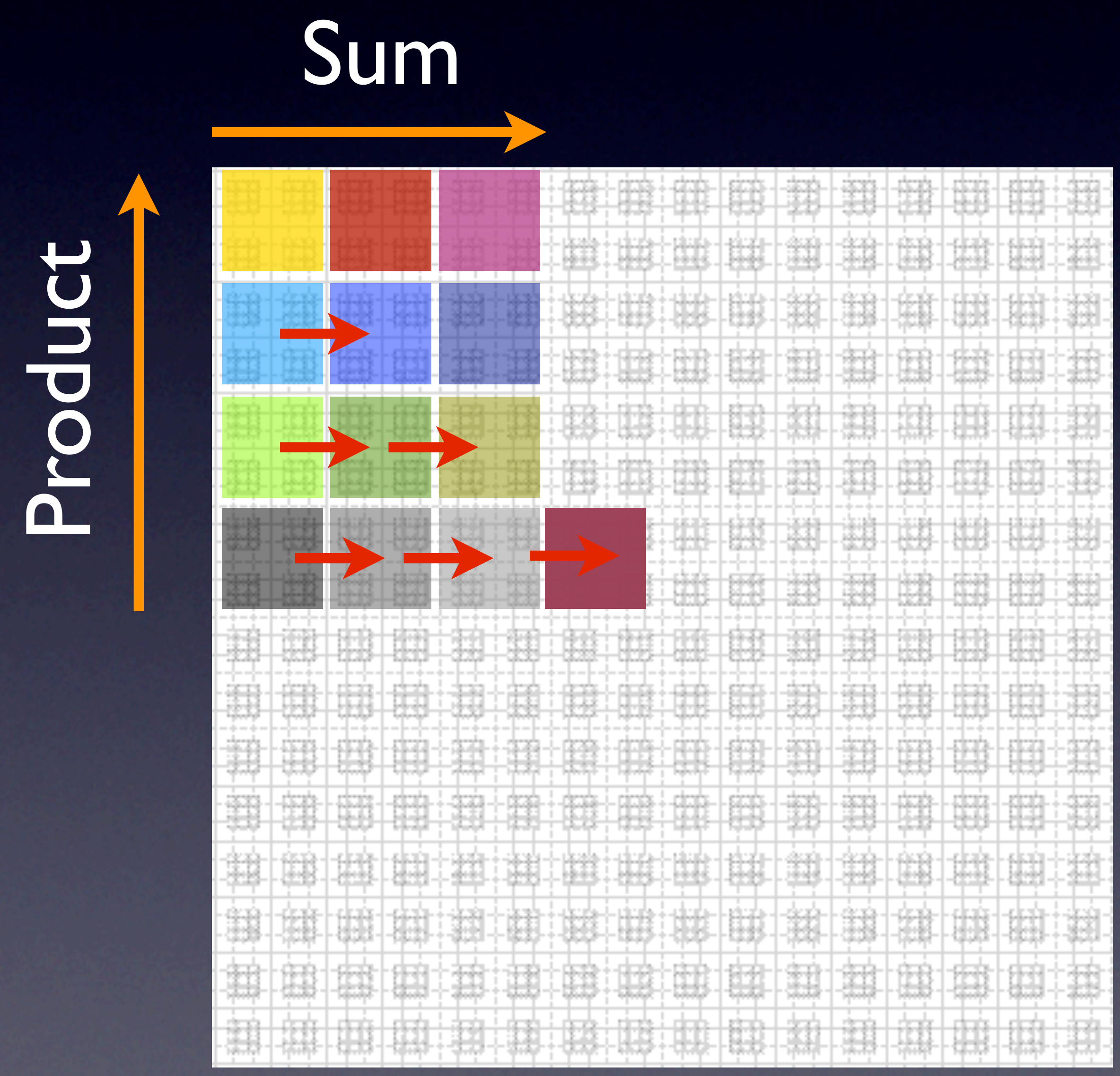
General Optimization Solvers in the Boolean CA

“Rendering” Math

Linear time Matrix-Vector Multiplication

$$A\,x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

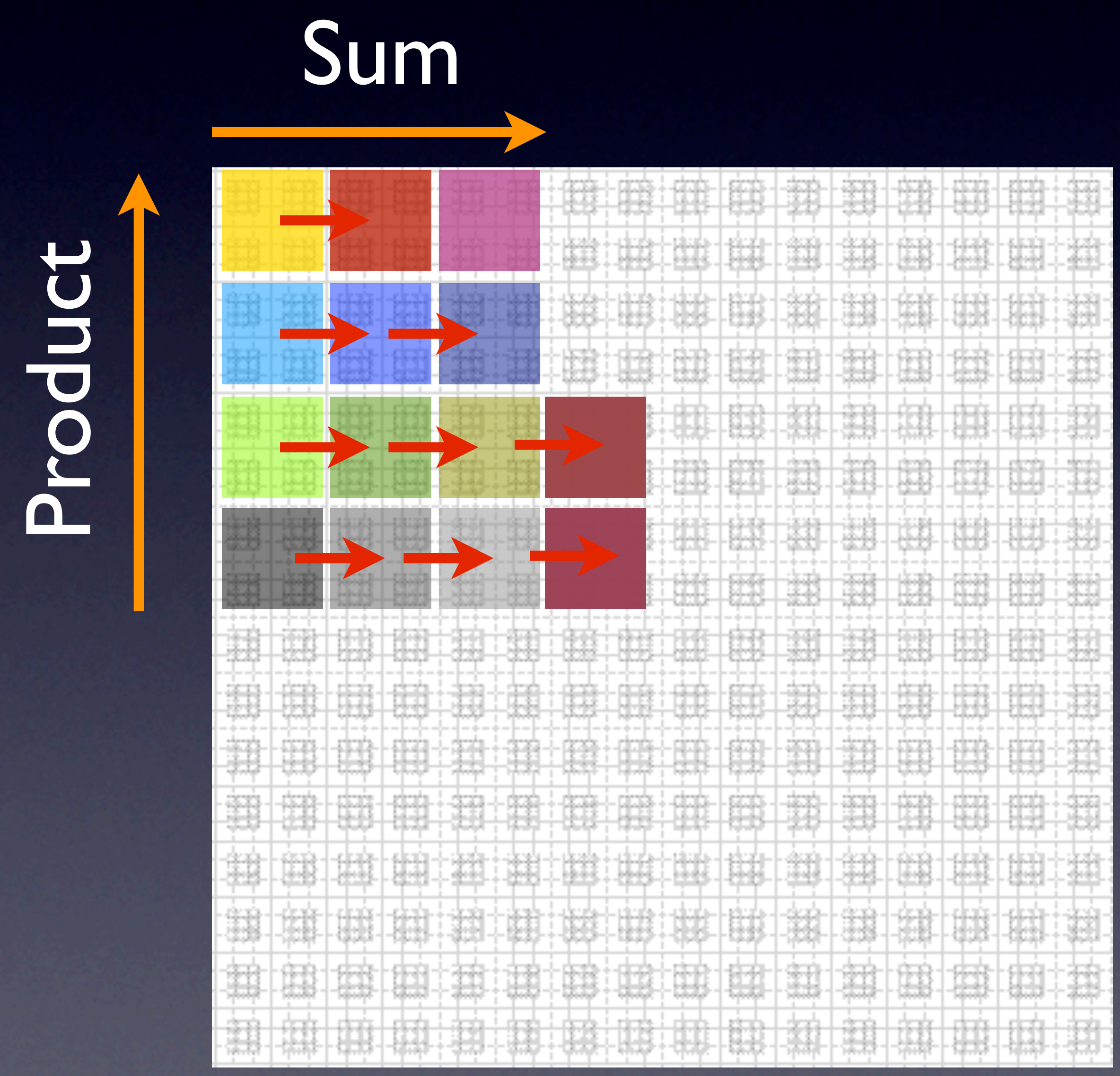


General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A\,x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$



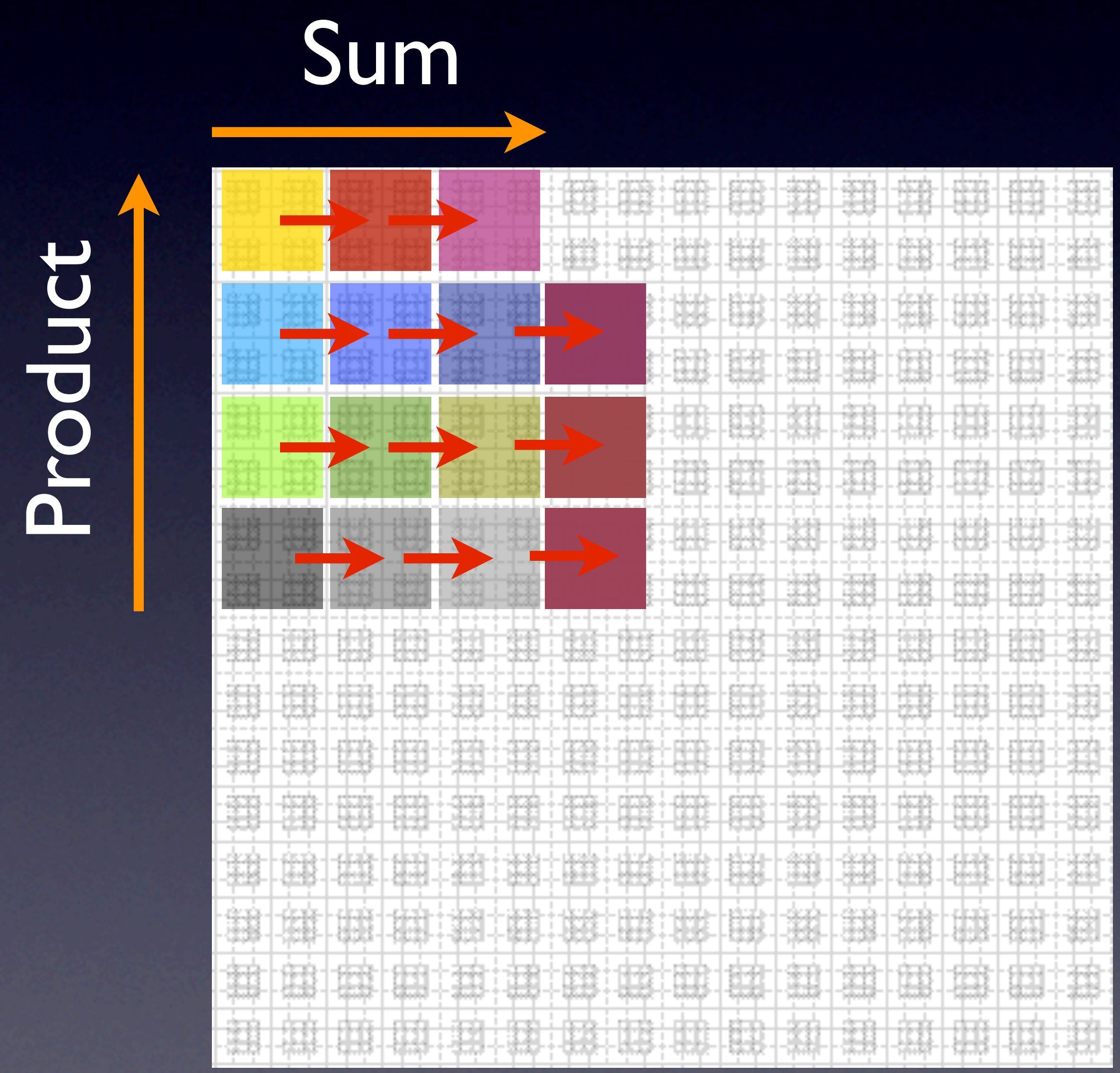
General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$
\begin{matrix} A \\ x \end{matrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}
$$

$$
\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}
$$



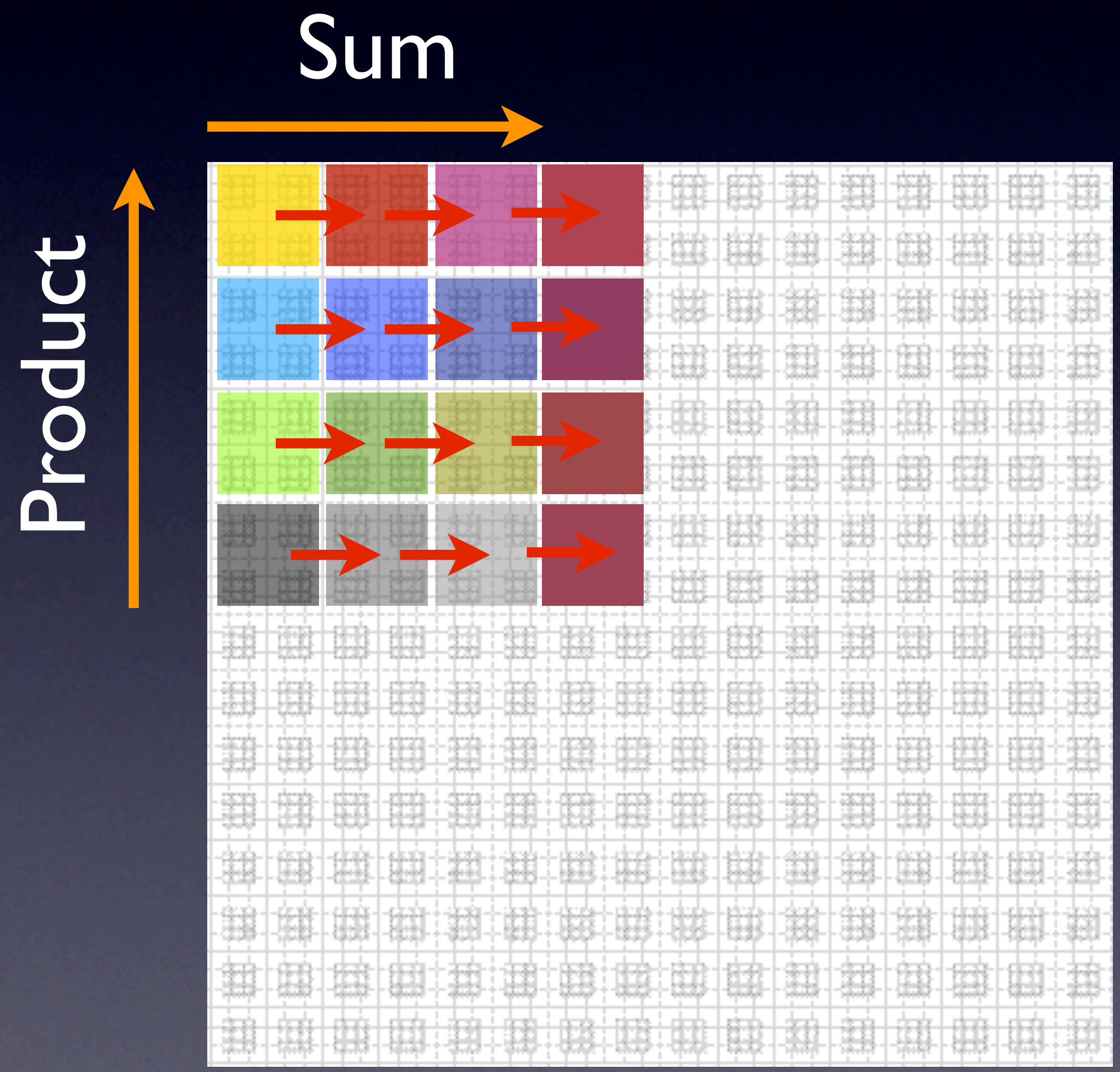
General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$

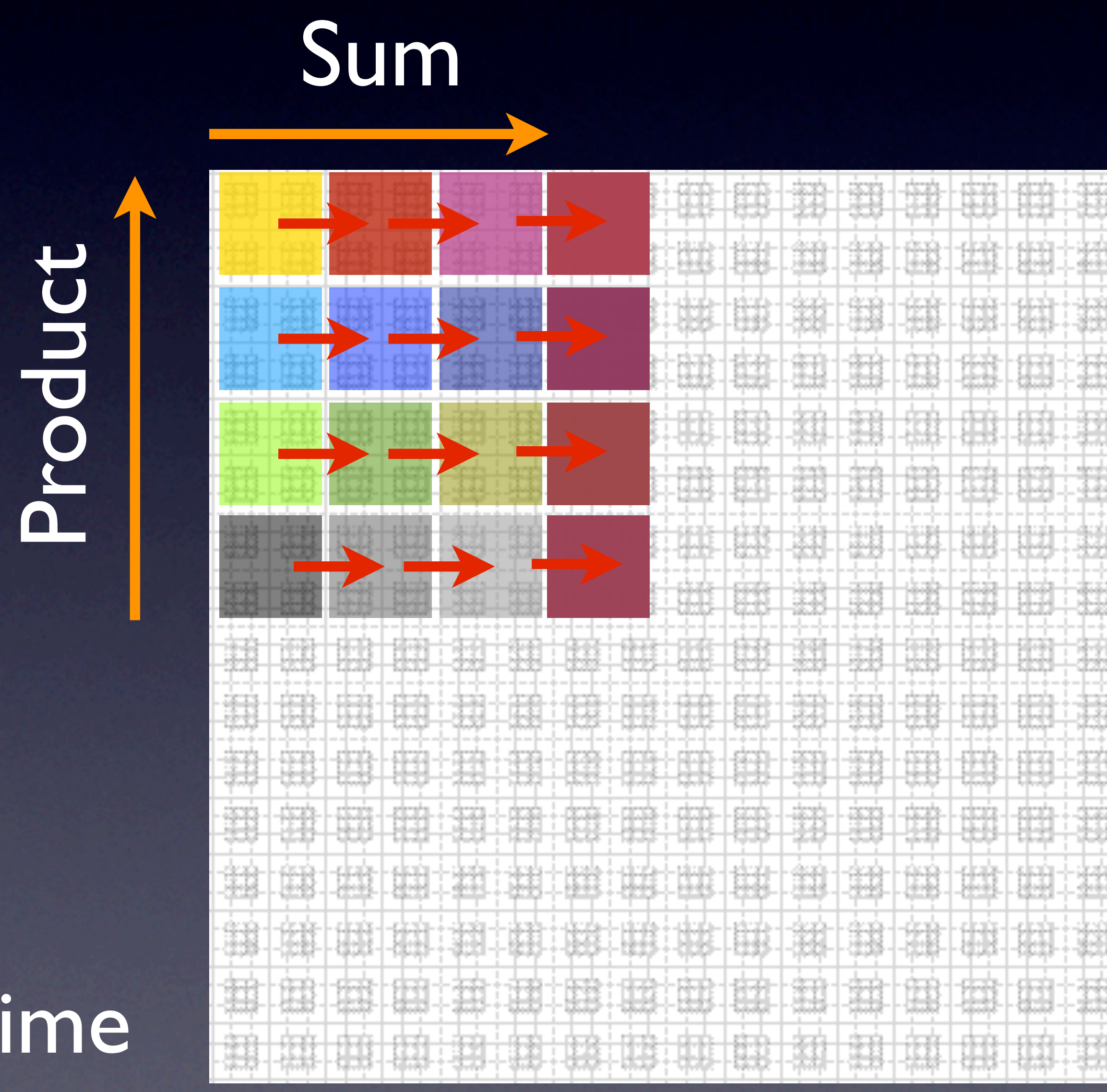


General Optimization Solvers in the Boolean CA

"Rendering" Math

Linear time Matrix-Vector Multiplication

$$A\,x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$$



$m \times n$ matrix ➡ $m + n$ time