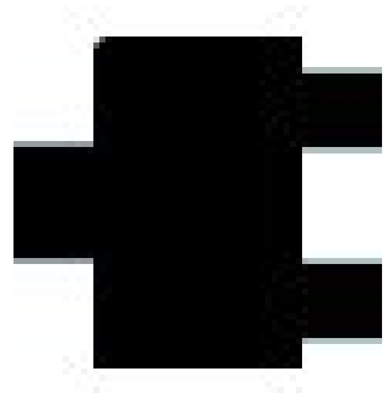
Nanophotonic Computational Design

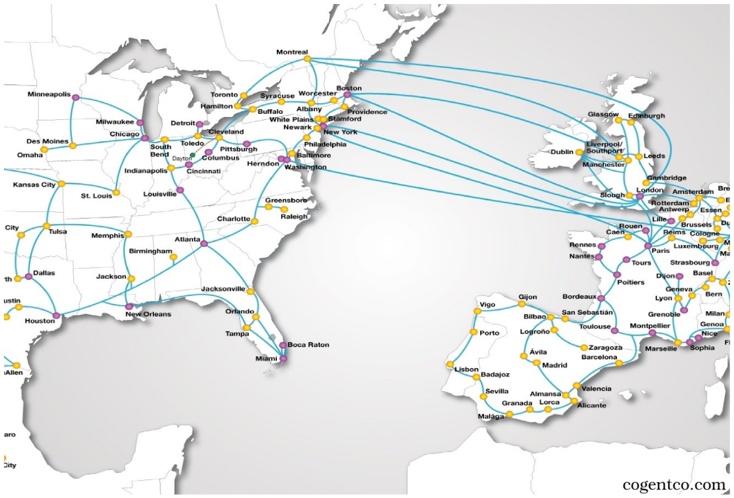
Jesse Lu

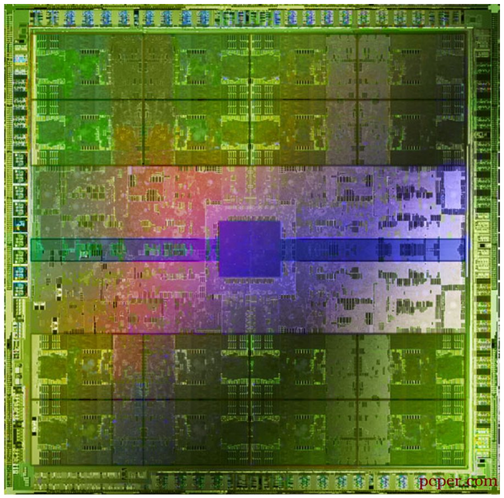
February 25, 2013

Takeaway: Taught a computer to design nanophotonic devices



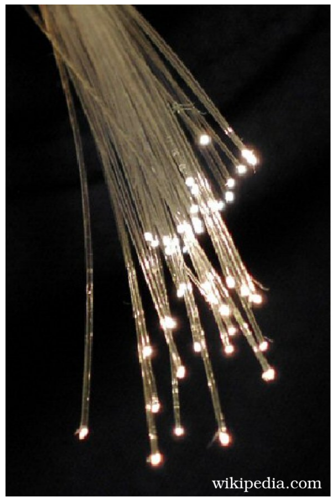
Part 1: Motivation

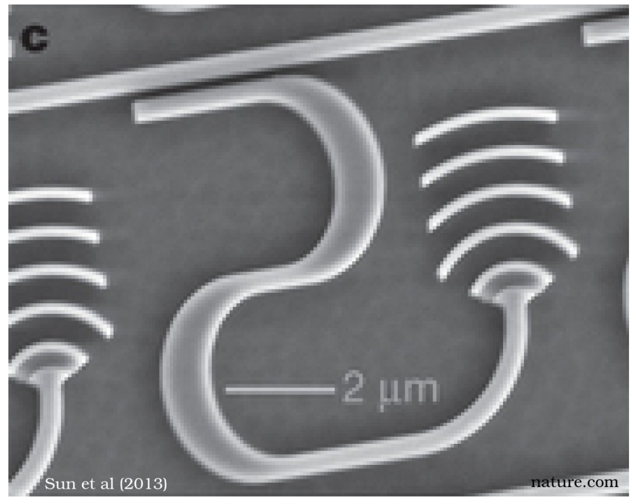




- As information grows, optical networks needed
  - across continents
  - within a datacenter
  - between chips and on-chip

- An on-chip optical network is a fundamentally new optical communications technology: *the integrated optical circuit*

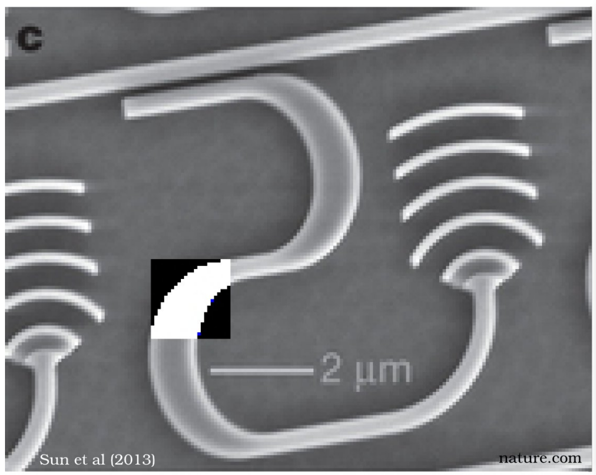


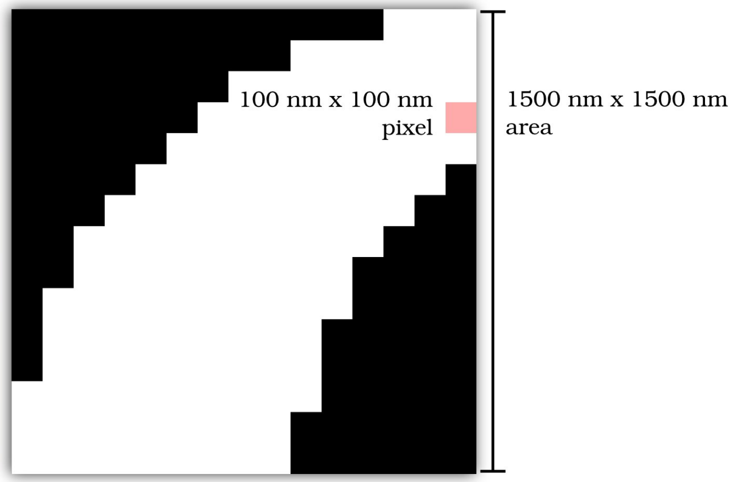


- Miniaturization drives
  - component price down
  - functionality up
  - design complexity (way) up

- Increasing design complexity requires additional degrees of freedom

- Fortunately, we have a virtually unlimited amount

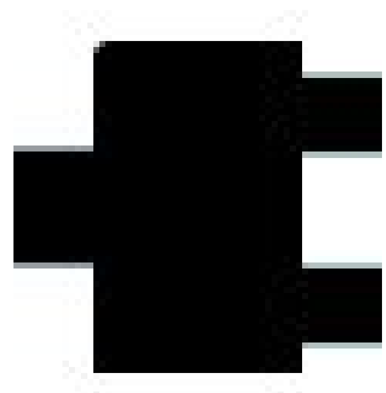




- Include/exclude per pixel gives us $2^{(15^2)} = 2^{225}$ possibilities, uncountable

- Only feasible solution: Humans describe, Computers design

```
device mux2
    in: {freq1, freq2}
    out1 <= freq1
    out2 <= freq2
```



Problem formulation:

$$\text{optimize} \quad f_{\mathsf{perf}}(H) + g_{\mathsf{manuf}}(\epsilon) \tag{1a}$$

$$\text{subject to} \quad \nabla \times \epsilon^{-1} \nabla \times H - \mu\omega^2 H = 0 \tag{1b}$$

In the language of linear algebra,

$$\text{minimize} \quad f(x) + g(p) \tag{2a}$$

$$\text{subject to} \quad r(x, p) = 0 \tag{2b}$$

- x is the field variable , p is the structure variable
- $f(x) + g(p)$ is the design objective
- $r(x, p)$ is the physics residual

Generic nonlinear optimization packages

- It is possible to efficiently compute the following:
  - Zeroth-order: f, g, r, and $\|r\|^2$
  - First-order: ∇f, ∇g, ∇r, and $\nabla \|r\|^2$
  - Second-order: $\nabla^2 f$, $\nabla^2 g$, and $\nabla^2 \|r\|^2$

- However, most efficient solvers require computing either $(\nabla r)^{-1} z$ or $(\nabla^2 \|r\|^2)^{-1})z$, which is often very difficult!

- Many viable approximations exist, but convergence is often slow and unreliable.

Key insights/assumptions

1. $r(x, p)$ is separably affine (bi-affine) in x and p,

$$r(x, p) = A(p)x - b(p) = B(x)p - d(x), \tag{3}$$

   this allows us to form two simpler sub-problems.

2. Simulators which compute $A(p)^{-1}z$ are available, even for very large systems (millions of variables).

3. Solving $B(x)p - d(x) = 0$ is possible, because manufacturing processes severely limit the degrees of freedom of p (decreases p to thousands of variables).

Adjoint method

- Starting at $r(x_0, p_0) = 0$, solves

$$\begin{aligned} \text{minimize} \quad & f(x) + g(p) && \text{(4a)} \\ \text{subject to} \quad & r(x,p) = 0 && \text{(4b)} \end{aligned}$$

by steepest descent along $\frac{df}{dp} + \frac{dg}{dp}$ while enforcing $r(x,p) = 0$.

- Computationally efficient because $\frac{df}{dp}$ is computed in a single simulation.

- A total of only two simulations required per iteration.

- Steepest-descent methods usually exhibit very slow convergence, but this method has proven very useful in practice, especially because $r(x,p) = 0$ at every iteration.

Alternating directions

- Alternatively, we can break our problem into two separate subproblems, taking advantage of

$$r(x,p) = A(p)x - b(p) = B(x)p - d(x). \tag{5}$$

- x and p are iteratively updated,

$$x := \operatorname*{argmin}_x f(x) + \frac{\rho}{2}\|Ax - b\|^2 \tag{6a}$$

$$p := \operatorname*{argmin}_p g(p) + \frac{\rho}{2}\|Bp - d\|^2. \tag{6b}$$

- Allows us to start from $r(x_0, p_0) \neq 0$ and then gradually increase ρ until $r(x,p) = 0$.

Alternating directions method of multipliers (ADMM)

- Include additional (dual) variable y,

$$x := \operatorname*{argmin}_x f(x) + \frac{\rho}{2}\|Ax - b\|^2 + y^T(Ax - b) \tag{7a}$$

$$p := \operatorname*{argmin}_p g(p) + \frac{\rho}{2}\|Bp - d\|^2 + y^T(Bp - d) \tag{7b}$$

$$y := y + \rho r(x, p) \tag{7c}$$

- Works for fixed ρ and generally exhibits faster convergence than alternating directions.

- We assume that updating x takes up the most computational resources.

- For what choices of $f(x)$ can we efficiently solve

$$\underset{x}{\operatorname{argmin}} L(x,p,y) \;=\; \underset{x}{\operatorname{argmin}} f(x) + \frac{\rho}{2}\|Ax - b\|^2 + y^T(Ax - b) \quad (8)$$

- Solve quadratic approximation (Newton's method),

$$\Delta x = (\nabla^2_{xx} L)^{-1} \nabla_x L, \quad (9)$$

where

$$\nabla_x L(x,p,y) = \nabla f(x) + A^T(\rho(Ax - b) + y) \quad (10a)$$

$$\nabla^2_{xx} L(x,p,y) = \nabla^2 f(x) + \rho A^T A \quad (10b)$$

- Assumption: for general $\nabla^2 f(x)$, $\nabla^2_{xx} L(x,p,y)$ *cannot* be inverted.

- Case 1: $f(x) = c^T x$, Field overlap integral
- In this case

$$(\nabla^2_{xx} L(x, p, y))^{-1} = \rho^{-1}(A^T A)^{-1} = \rho^{-1} A^{-1} A^{-T} \tag{11a}$$

$$\nabla_x L(x, p, y) = c + A^T(\rho(Ax - b) + y), \tag{11b}$$

so we can solve

$$\Delta x = (\nabla^2_{xx} L)^{-1} \nabla_x L \tag{12a}$$

$$= \rho^{-1} A^{-1}(A^{-T} c + \rho(Ax - b) + y) \tag{12b}$$

using only two simulations.

- Since $L(x, p, y)$ is exactly quadratic, we can update x using only two simulations.

- Case 2: $f(x) = \|C^T x\|^2$, Energy in mode

- Case 3: $f(x)$ forces $C^T x = d$.

- Also, multi-mode.