Linear Programming and DFM

- Course contents:
  - Linear Programming
  - Integer Linear Programming
  - Design for Manufacturability (DFM)
- Reference
  - Cormen, *Introductions to Algorithms*, 3rd Ed.
  - Bradley, *Applied Mathematical Programming*
  - Research papers

Linear Programming and Integer Linear Programming

Linear Programming

- Linear programming describes a broad class of optimization tasks in which both the optimization objective and the constraints are linear functions

- Linear programming consists of three parts:
  - A set of decision variables
  - An objective function
    - Maximize or minimize a given linear objective function
  - A set of constraints
    - Satisfy a set of linear inequalities involving these variables

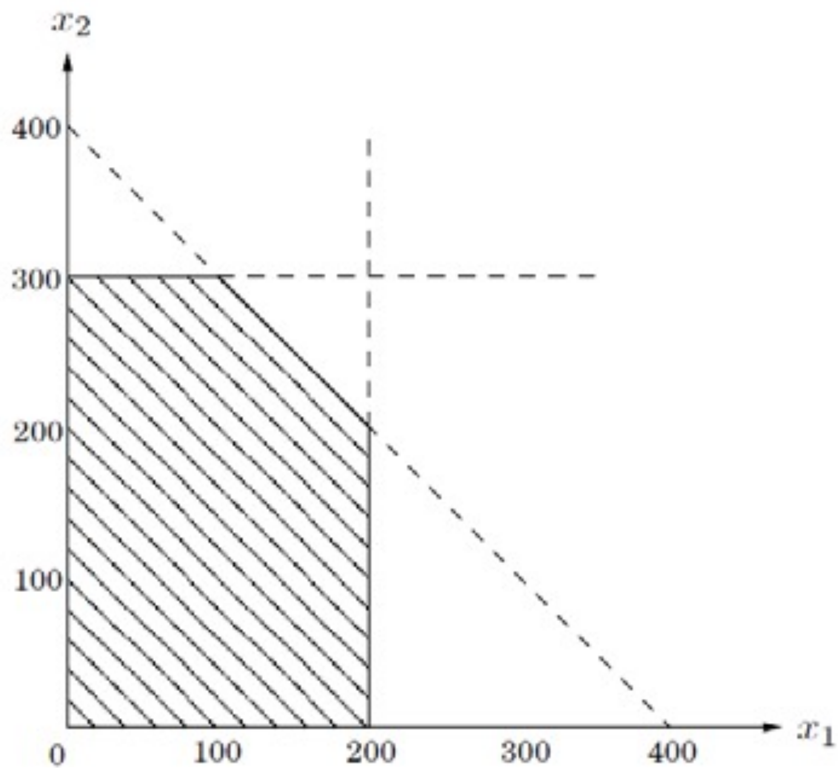
An Example

- A boutique chocolatier has two products:
  - A product: profit \$1 per box
  - B product: profit \$6 per box
- Constraints
  - The daily demand for these exclusive chocolates is limited to at most 200 boxes of A and 300 boxes of B
  - The current workforce can produce a total of at most 400 boxes of chocolate per day
- Decision variables
  - x_1: #boxes of A
  - x_2: #boxes of B
- Objective Function
  - Maximize profit

$$
\begin{array}{ll}
\text{Maximize} & x_1 + 6x_2 \\
\text{Subject to} & x_1 \le 200 \\
 & x_2 \le 300 \\
 & x_1 + x_2 \le 400 \\
 & x_1, x_2 \ge 0
\end{array}
$$

An Example (cont'd)

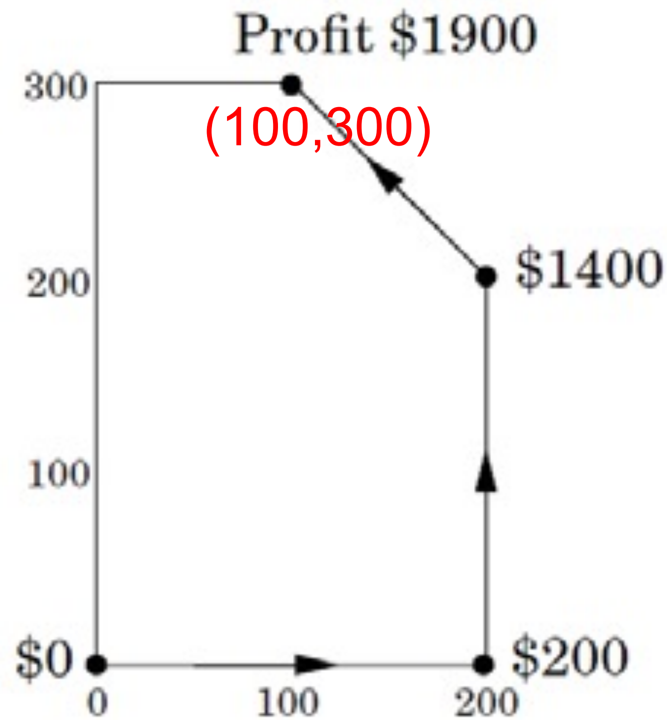
- A linear equation in x1 and x2 defines a line in the 2D plane
- A linear inequality designates a half-space
- The set of all feasible solutions of this linear program is the intersection of five half-spaces, which is a convex polygon



Maximize	$x_1 + 6x_2$
Subject to	$x_1 \leq 200$
	$x_2 \leq 300$
	$x_1 + x_2 \leq 400$
	$x_1, x_2 \geq 0$

An Example (cont'd)

- The Simplex method to find the optimal solution
  - Starts at a vertex, say (0, 0)
  - Repeatedly looks for an adjacent vertex (connected by an edge of the feasible region) that has better objective value
  - Reaching a vertex with no better neighbor, it is the optimal solution and Simplex terminates



General Linear Programs

- Standard form

Maximize $$\sum_{j=1}^{n} c_j \cdot x_j$$

Subject to $$\sum_{j=1}^{n} a_{ij} \cdot x_j \leq b_i, \forall\, i = 1,2, \dots, m$$

$$x_j \geq 0, \forall\, j = 1,2, \dots, n$$

- It is easy to convert a linear program into standard form
  - A minimization objective
  - Variables without nonnegativity constraints
  - Program with equality constraints
  - Program with greater-than-or-equal-to inequality constraints

Integer Linear Programming (ILP)

- The linear-programming models are continuous
  - Decision variables are allowed to be fractional
- When fraction solutions are not allowed

Maximize $$\sum_{j=1}^{n} c_j \cdot x_j$$

Subject to $$\sum_{j=1}^{n} a_{ij} \cdot x_j \leq b_i, \forall\, i = 1,2, \dots, m$$

$$x_j \geq 0, \forall\, j = 1,2, \dots, n$$

x_j is integer, for some or all $j = 1,2, \dots, n$

  - This is called (mixed) integer linear programming

Binary Variables

- To model yes-no decision in an ILP formulation, use binary (0-1) variables
- An ILP where all variables are binary is a 0-1 ILP formulation
  - Usually, can be solved more efficiently than general ILP

Maximize $\sum_{j=1}^{n} c_j \cdot x_j$

Subject to $\sum_{j=1}^{n} a_{ij} \cdot x_j \leq b_i, \forall\, i = 1,2, \dots, m$

$x_j = 0 \; or \; 1, \forall\, j = 1,2, \dots, n$

TSP with 0/1-ILP

- Starting from his/her home, a salesman visits each of (n −1) other cities exactly once and return home at minimal cost
  - x_{ij}: binary variable; $x_{ij} = 1$ if he goes from city i to city j; $x_{ij} = 0$ ootherwise
  - c_{ij}: the cost goes from city i to city j

Maximize $$\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}$$

Subject to $$\sum_{i=1}^{n} x_{ij} = 1, \forall\, j = 1,2, \ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \forall\, i = 1,2, \ldots, n$$

$$x_{ij} = 0\ or\ 1, \forall\, j = 1,2, \ldots, n$$

Logical Constraints

- Constraint feasibility
  - To see when is a general constraint satisfied
  - To relax some constraints such that they may not be satisfied

$$f(x_1, x_2, \dots, x_n) \le b \quad \Longrightarrow \quad f(x_1, x_2, \dots, x_n) \le b + By$$

$$y = \begin{cases} 0, & \text{the constraint is satisifed} \\ 1, & \text{otherwise} \end{cases}$$

  - B is chosen to be large enough so that the constraint always is satisfied if $y = 1$

Logical Constraints (cont'd)

- Alternative constraints
  - At least one of two constraints must be satisfied

$$f_1(x_1, x_2, \ldots, x_n) \le b_1 \qquad\qquad f_1(x_1, x_2, \ldots, x_n) \le b_1 + By_1$$

$$f_2(x_1, x_2, \ldots, x_n) \le b_2 \quad \Longrightarrow \quad f_2(x_1, x_2, \ldots, x_n) \le b_2 + By_2$$

At least one of f_1 and f_2 must be satisfied

$$y_1 + y_2 \le 1$$

$$y_1, y_2 = 0\ or\ 1$$

  - B is chosen to be large enough so that the constraint always is satisfied if y_1 or $y_2 = 1$

Logical Constraints (cont'd)

- Conditional constraint
  - If Constraint 1 is satisfied, then Constraint 2 must be satisfied

    $$\text{if } f_1(x_1, x_2, \dots, x_n) > b_1, \text{then } f_2(x_1, x_2, \dots, x_n) \leq b_2$$

  - The implication is not satisfied only when

    $$f_1(x_1, x_2, \dots, x_n) > b_1, \text{and } f_2(x_1, x_2, \dots, x_n) > b_2$$

  - The conditional constraint can be modeled by the alternative constraint

    $$f_1(x_1, x_2, \dots, x_n) \leq b_1, \text{and/or } f_2(x_1, x_2, \dots, x_n) \leq b_2$$

Logical Constraints (cont'd)

- k-fold alternatives
  - Must satisfy at least k of the constraints:

$$f_j(x_1, x_2, \dots, x_n) \le b_j, \qquad j = 1,2,\dots,p$$

$$\Downarrow$$

$$f_j(x_1, x_2, \dots, x_n) \le b_j + B_j(1 - y_j)$$

$$\sum_{j=1}^{p} y_j \ge k$$

$$y_j = 0 \ or \ 1, j = 1,2,\dots,p$$

  - $y_j = 1$ if Constraint j is satisfied

Logical Constraints (cont'd)

- Compound alternatives
  - The feasible solution space consists of three disjoint regions, each specified by a system of inequalities

$$\left.\begin{aligned} f_1(x_1, x_2) &\le b_1 + B_1 y_1 \\ f_2(x_1, x_2) &\le b_2 + B_2 y_1 \end{aligned}\right\} \text{Region 1}$$

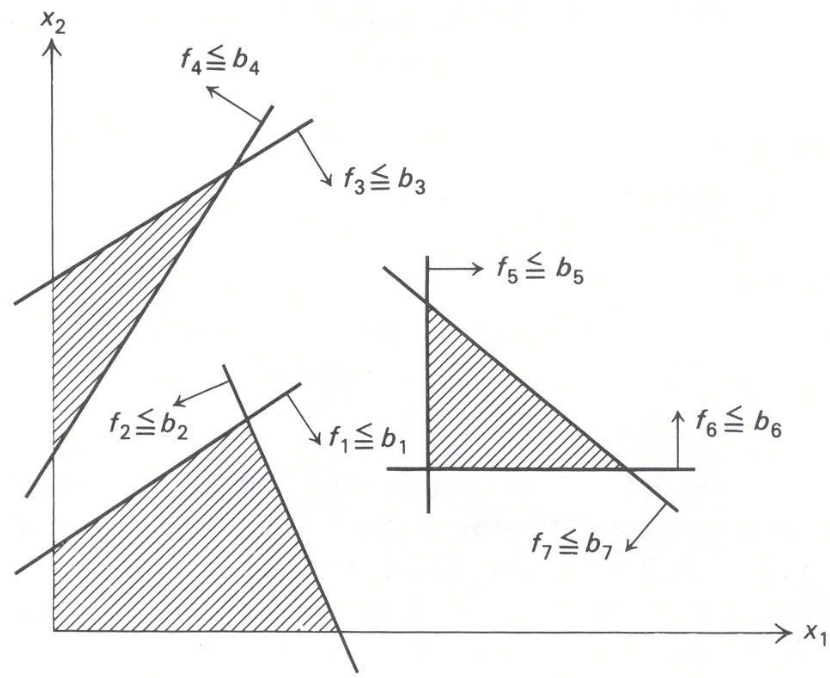
$$\left.\begin{aligned} f_3(x_1, x_2) &\le b_3 + B_3 y_2 \\ f_4(x_1, x_2) &\le b_4 + B_4 y_2 \end{aligned}\right\} \text{Region 2}$$

$$\left.\begin{aligned} f_5(x_1, x_2) &\le b_5 + B_5 y_3 \\ f_6(x_1, x_2) &\le b_6 + B_6 y_3 \\ f_7(x_1, x_2) &\le b_7 + B_7 y_3 \end{aligned}\right\} \text{Region 3}$$

$$y_1 + y_2 + y_3 \le 2$$

$$x_1 \ge 0, x_2 \ge 0$$

$$y_1, y_2, y_3 = 0 \text{ or } 1$$



Representing Nonlinear Functions

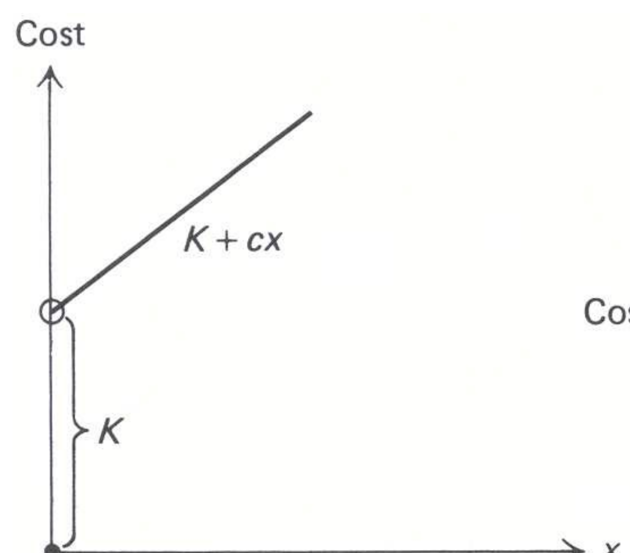
- Fixed costs
  - Frequently, the objective function for a minimization problem contains fixed costs

Minimize $Ky + cx$

Subject to $x \leq By$

$x \geq 0$

$y = 0\ or\ 1$



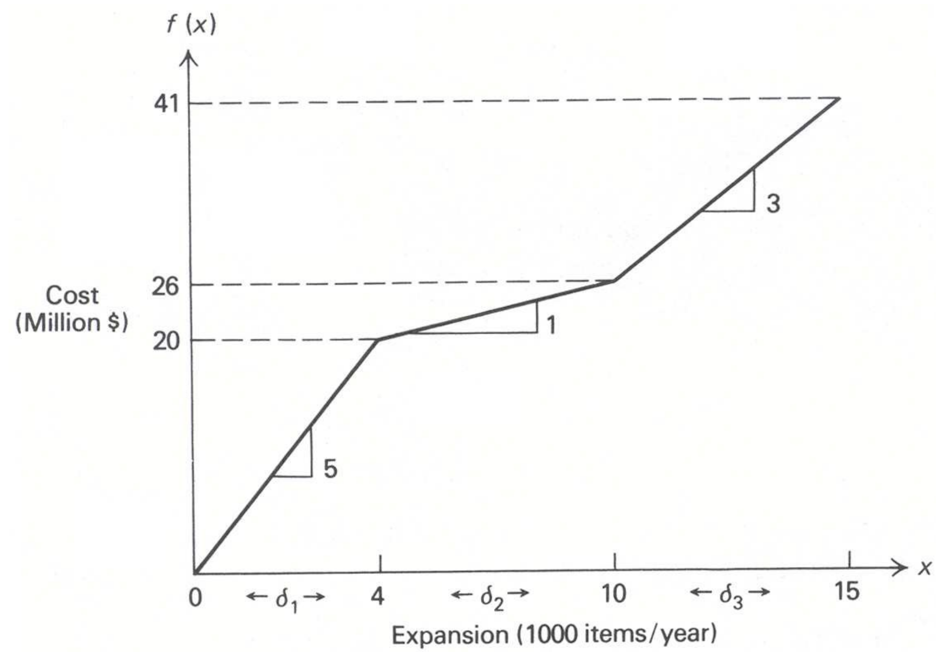
$$\text{Cost} = \begin{cases} 0 & \text{if } x = 0 \\ K + cx & \text{if } x > 0 \end{cases}$$

Representing Nonlinear Functions (cont'd)

- Piecewise linear
  - The cost is computed by a piecewise-linear function

$$cost = 5\delta_1 + 1\delta_2 + 3\delta_3$$
$$0 \le \delta_1 \le 4$$
$$0 \le \delta_2 \le 6$$
$$0 \le \delta_3 \le 5$$

  - Additional constraints
    - If $\delta_2 > 0, \Longrightarrow \delta_1 = 4$
    - If $\delta_3 > 0, \Longrightarrow \delta_2 = 6$



Representing Nonlinear Functions (cont'd)

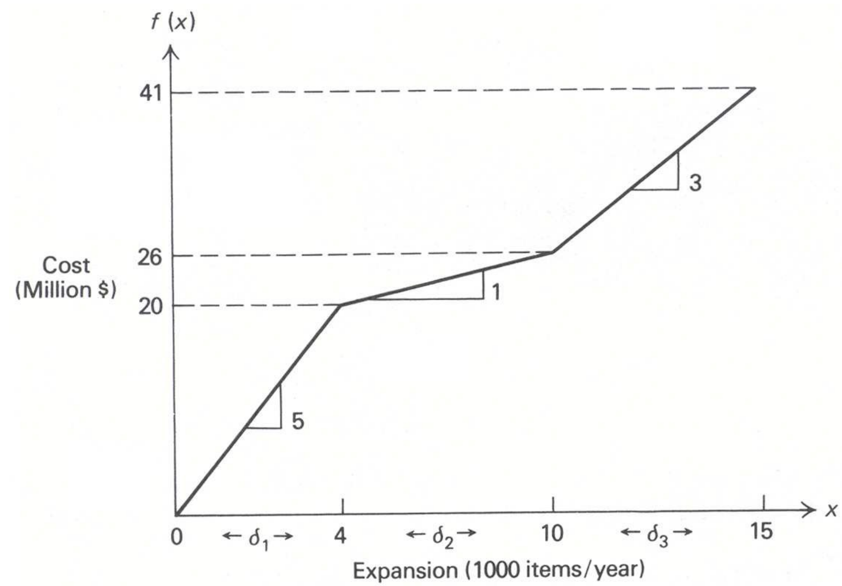
- Piecewise linear
  - The cost is computed by a piecewise-linear function

$$cost = 5\delta_1 + 1\delta_2 + 3\delta_3$$

$$w_1 = \begin{cases} 1, & if\ \delta_1\ is\ at\ its\ upper\ bound \\ 0, & otherwise \end{cases}$$

$$w_2 = \begin{cases} 1, & if\ \delta_2\ is\ at\ its\ upper\ bound \\ 0, & otherwise \end{cases}$$

$$4w_1 \le \delta_1 \le 4,$$
$$6w_2 \le \delta_2 \le 6w_1,$$
$$0 \le \delta_3 \le 5w_2,$$
$$w_1, w_2 = 0\ or\ 1$$



Representing Nonlinear Functions (cont'd)

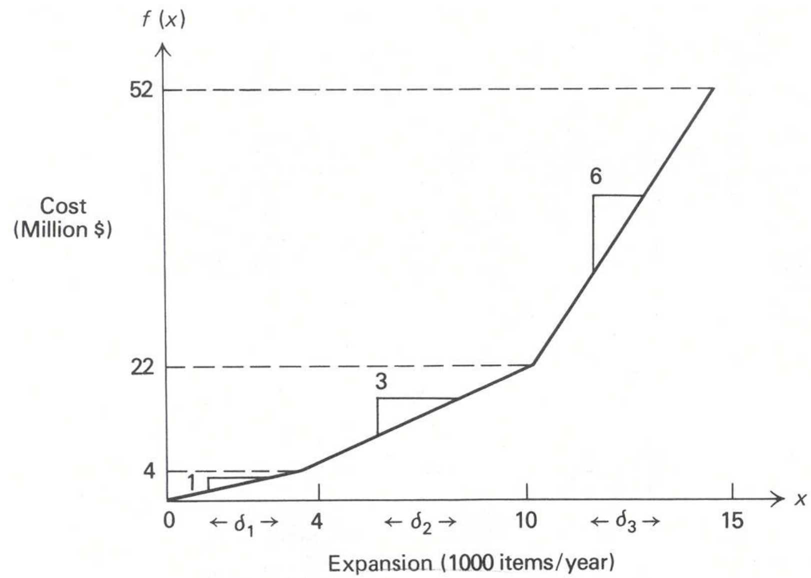
- When marginal costs are increasing for a minimization problem or vice versa, no integer variable is required
  - The cost of the piecewise-linear function in the figure:

    $cost = \delta_1 + 3\delta_2 + 6\delta_3$

    $0 \le \delta_1 \le 4$

    $0 \le \delta_2 \le 6$

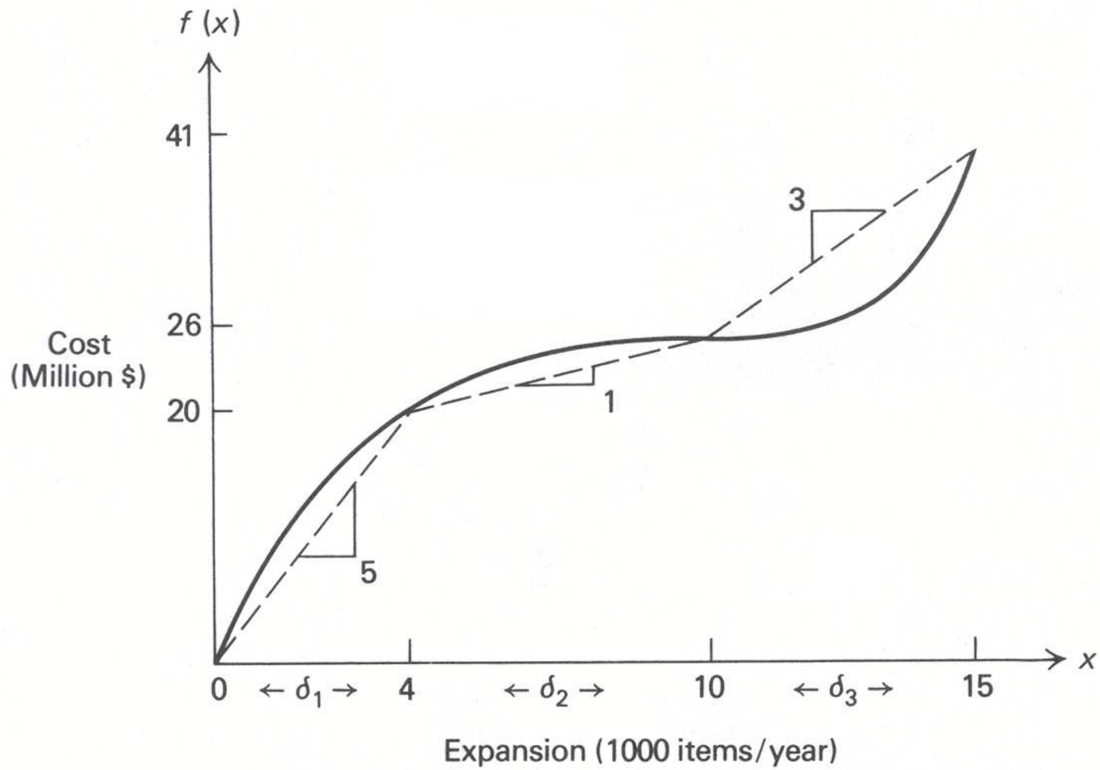
    $0 \le \delta_3 \le 5$



  - It is always better to set $\delta_1 = 4$ before taking $\delta_2 > 0$, and to set $\delta_2 = 6$ before taking $\delta_3 > 0$

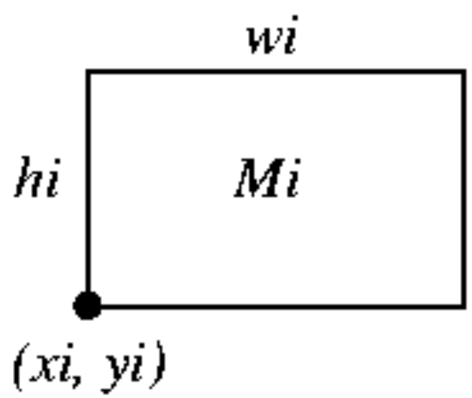
Representing Nonlinear Functions (cont'd)

- Approximation of nonlinear functions
  - One of the most useful applications of the piecewise linear representation is for approximating nonlinear functions



Floorplanning by Mathematical Programming

- Sutanthavibul, Shragowitz, and Rosen, "An analytical approach to floorplan design and optimization," 27th DAC, 1990.
- Notation:
  - w_i, h_i: width and height of module M_i.
  - (x_i, y_i): coordinate of the lower left corner of module M_i.
- Goal: Find a mixed **integer linear programming (ILP)** formulation for the floorplan design.
  - ***Linear*** constraints? Objective function?



Nonoverlap Constraints

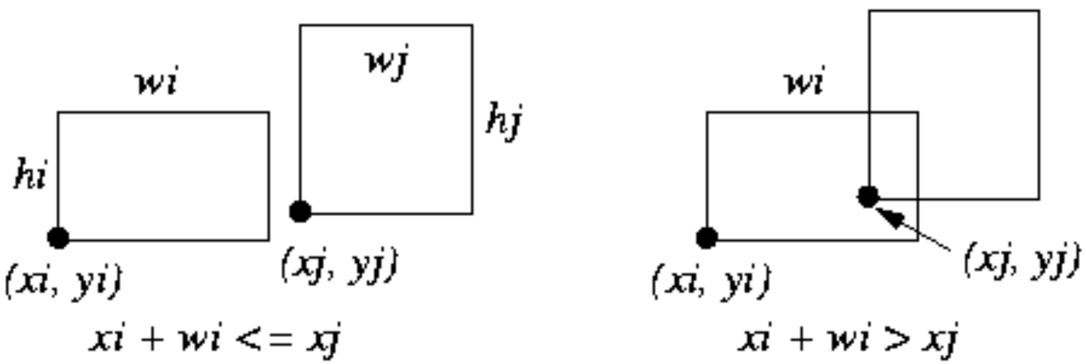
- Two modules M_i and M_j are nonoverlap, if at least one of the following linear constraints is satisfied
  - Encode each case by p_{ij} and q_{ij}

		p_{ij}	q_{ij}
M_i to the left of M_j:	$x_i + w_i \leq x_j$	0	0
M_i below M_j:	$y_i + h_i \leq y_j$	0	1
M_i to the right of M_j:	$x_i - w_j \geq x_j$	1	0
M_i above M_j:	$y_i - h_j \geq y_j$	1	1

Nonoverlap Constraints (cont'd)

- Let W, H be upper bounds on the floorplan width and height
- Introduce two 0, 1 variables p_{ij} and q_{ij} to denote that one of the following inequalities is enforced
  - E.g., $p_{ij} = 0$, $q_{ij} = 1 \Rightarrow$ $y_i + h_i \leq y_j$ is satisfied

$$
\begin{aligned}
x_i + w_i &\leq x_j + W(p_{ij} + q_{ij}) \\
y_i + h_i &\leq y_j + H(1 + p_{ij} - q_{ij}) \\
x_i - w_j &\geq x_j - W(1 - p_{ij} + q_{ij}) \\
y_i - h_j &\geq y_j - H(2 - p_{ij} - q_{ij})
\end{aligned}
$$



Cost Function & Constraints

- Minimize *Area* = xy, **nonlinear!** (x, y: width and height of the resulting floorplan)
- How to fix?
  - Fix the width W and minimize the height y!
- Four types of constraints:
  1. no two modules overlap ($\forall\ i, j$: $1 \le i < j \le n$);
  2. each module is enclosed within a rectangle of width W and height H ($x_i + w_i \le W$, $y_i + h_i \le H$, $1 \le i \le n$);
  3. $x_i \ge 0$, $y_i \ge 0$, $1 \le i \le n$;
  4. p_{ij}, $q_{ij} \in \{0, 1\}$.
- w_i, h_i are known.

Mixed ILP for Floorplanning

Mixed ILP for the floorplanning problem with rigid, fixed modules.

$$\min \quad y$$

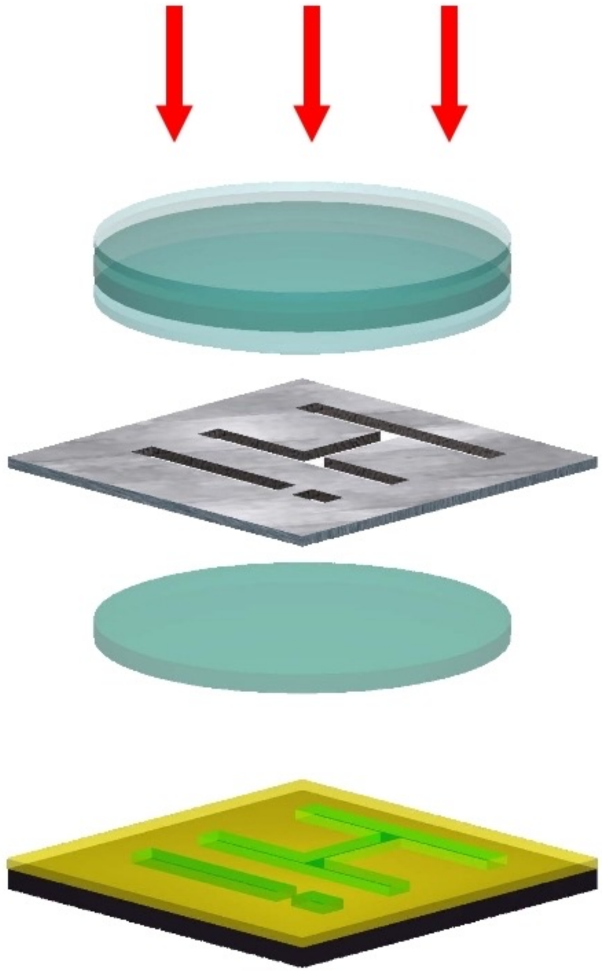
subject to

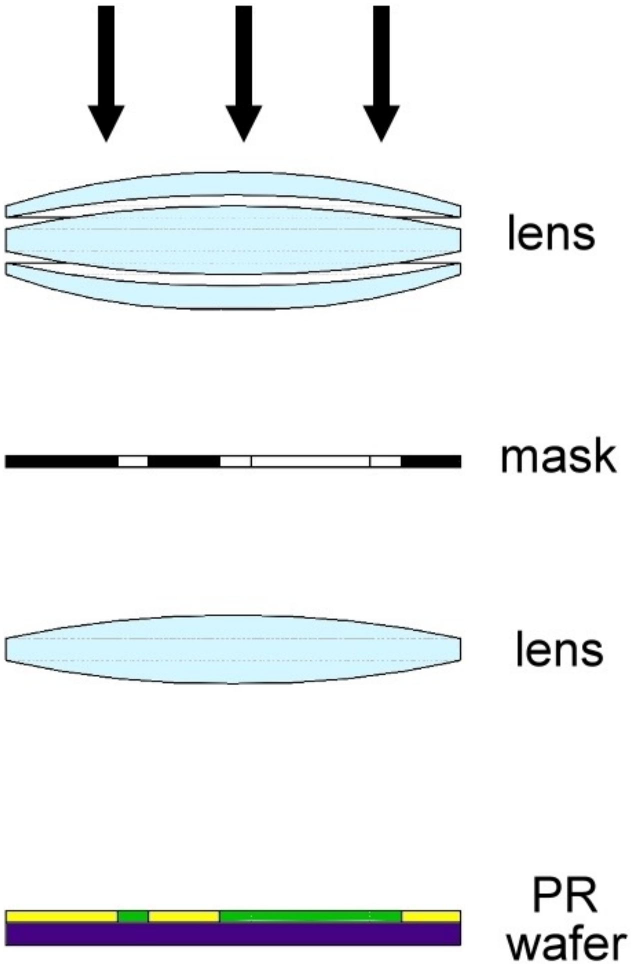
$$
\begin{aligned}
x_i + w_i &\le W, & 1 \le i \le n \quad &(1)\\
y_i + h_i &\le y, & 1 \le i \le n \quad &(2)\\
x_i + w_i &\le x_j + W(p_{ij} + q_{ij}), & 1 \le i < j \le n \quad &(3)\\
y_i + h_i &\le y_j + H(1 + p_{ij} - q_{ij}), & 1 \le i < j \le n \quad &(4)\\
x_i - w_j &\ge x_j - W(1 - p_{ij} + q_{ij}), & 1 \le i < j \le n \quad &(5)\\
y_i - h_j &\ge y_j - H(2 - p_{ij} - q_{ij}), & 1 \le i < j \le n \quad &(6)\\
x_i, y_i &\ge 0, & 1 \le i \le n \quad &(7)\\
p_{ij}, q_{ij} &\in \{0, 1\}, & 1 \le i < j \le n \quad &(8)
\end{aligned}
$$

- Popular LP software: LINDO, lp_solve, CPLEX, etc.

Design for Manufacturability (DFM)

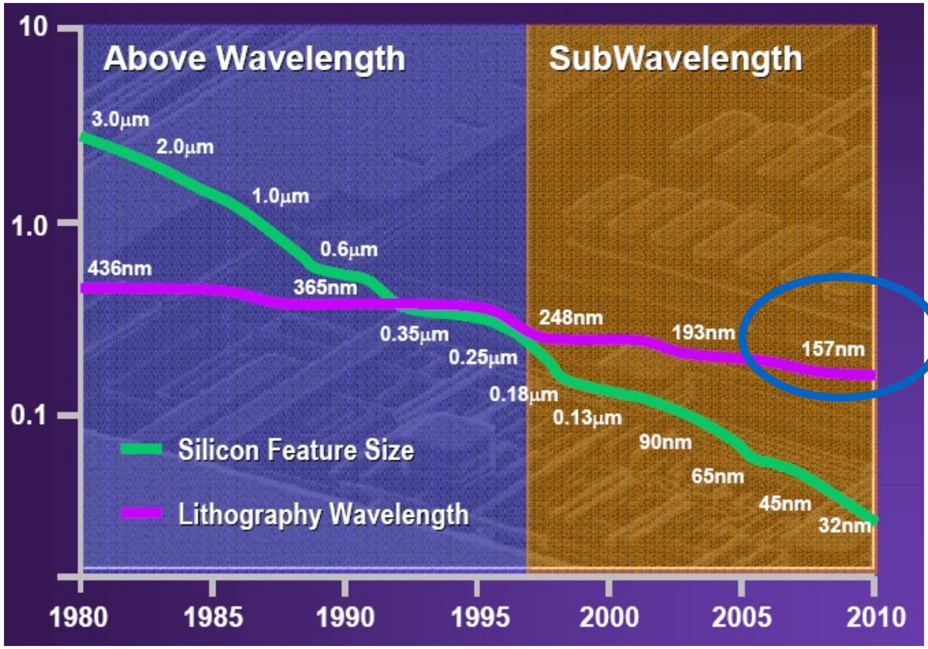
Basic Lithography System





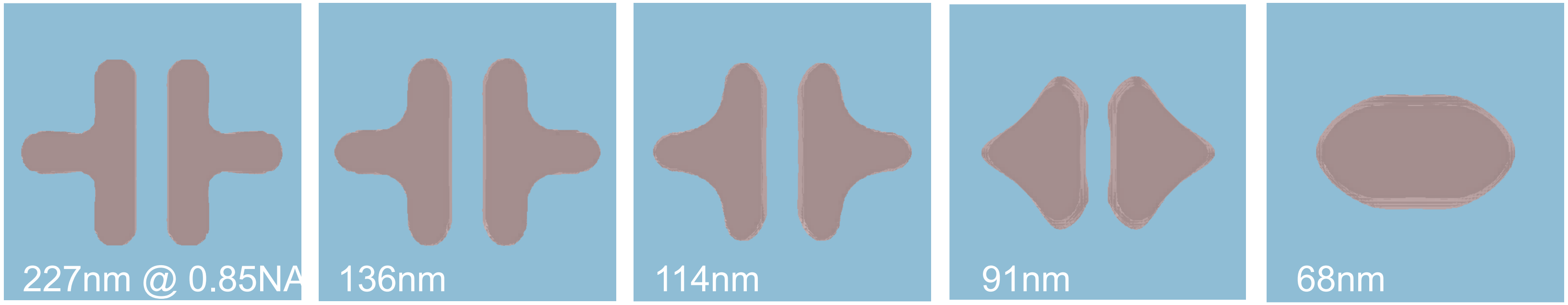
Subwavelength Lithography Gap

- Printed feature size is smaller than the wavelength of the light shining through the mask



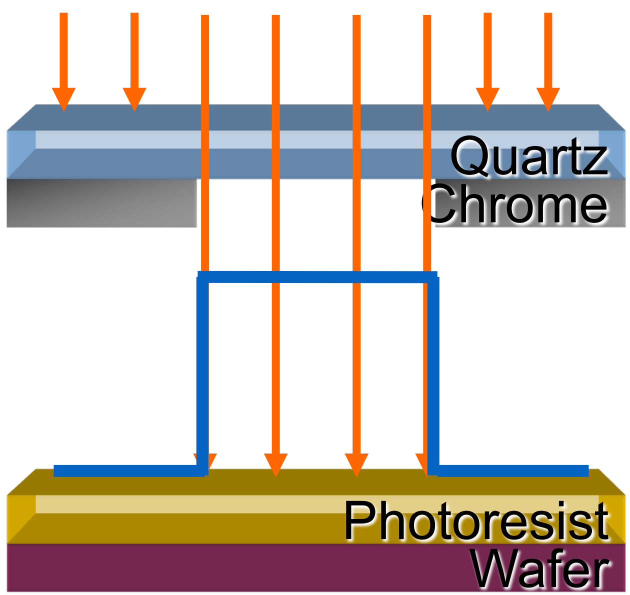
157nm will not be feasible!!

(go for EUV!!)

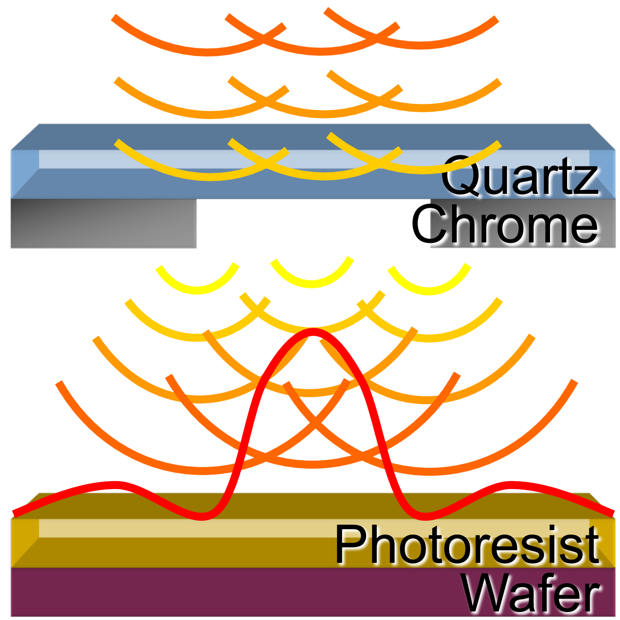


Optical Effect

- Ideal lithography: light passes through features by a straight path
- Real lithography: light behaves like waves when feature size is close to wavelength



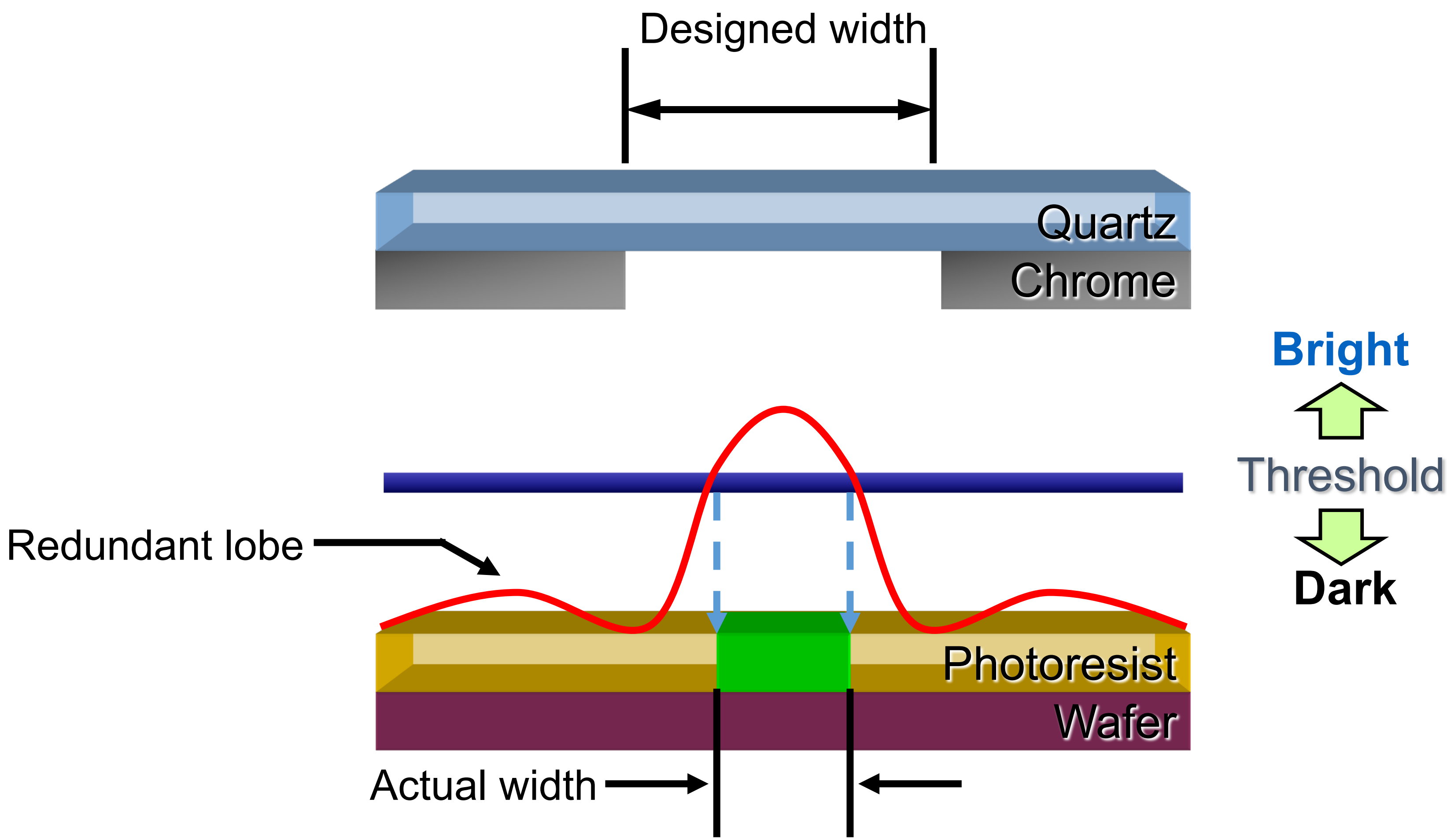
ideal lithography



real lithography

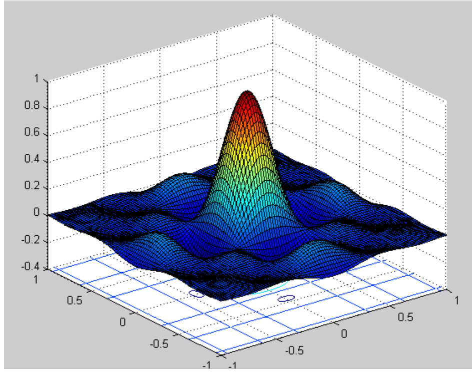
Optical Effect (cont'd)

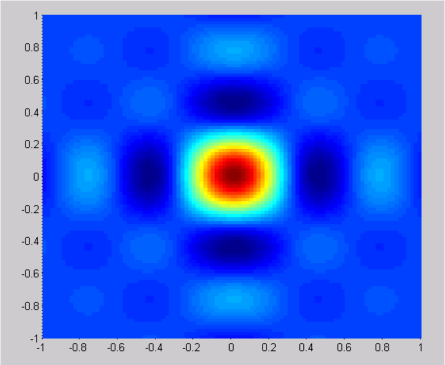
- Real lithography:



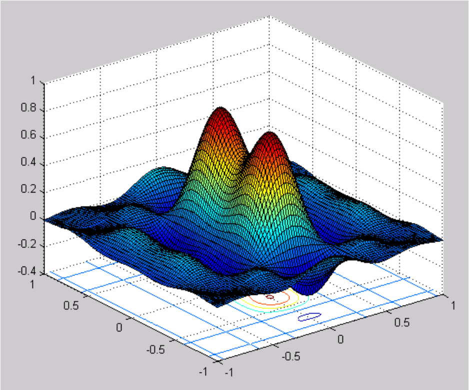
Simulation Results

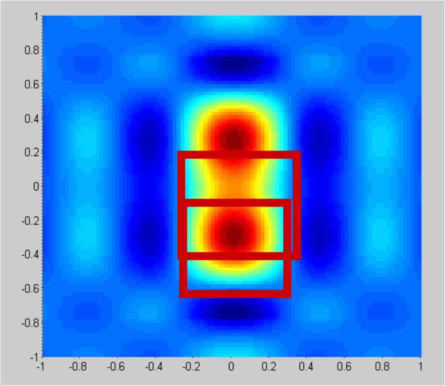
- Diffraction pattern of a single aperture





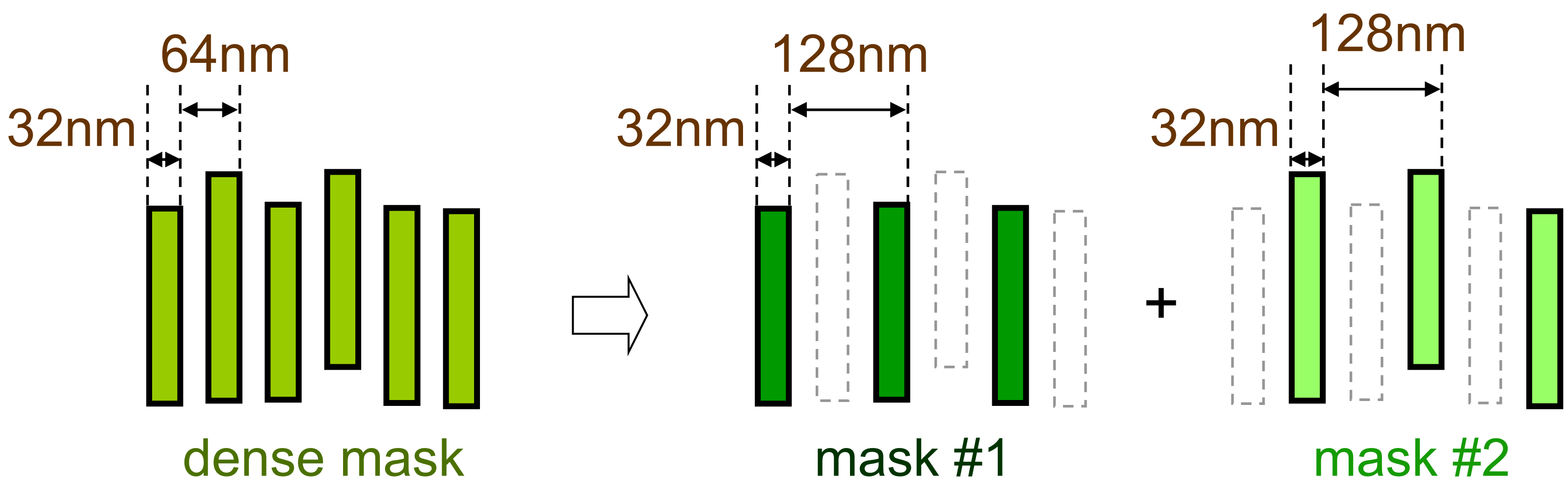
- Bridged diffraction pattern





Double-Patterning Lithography (DPL)

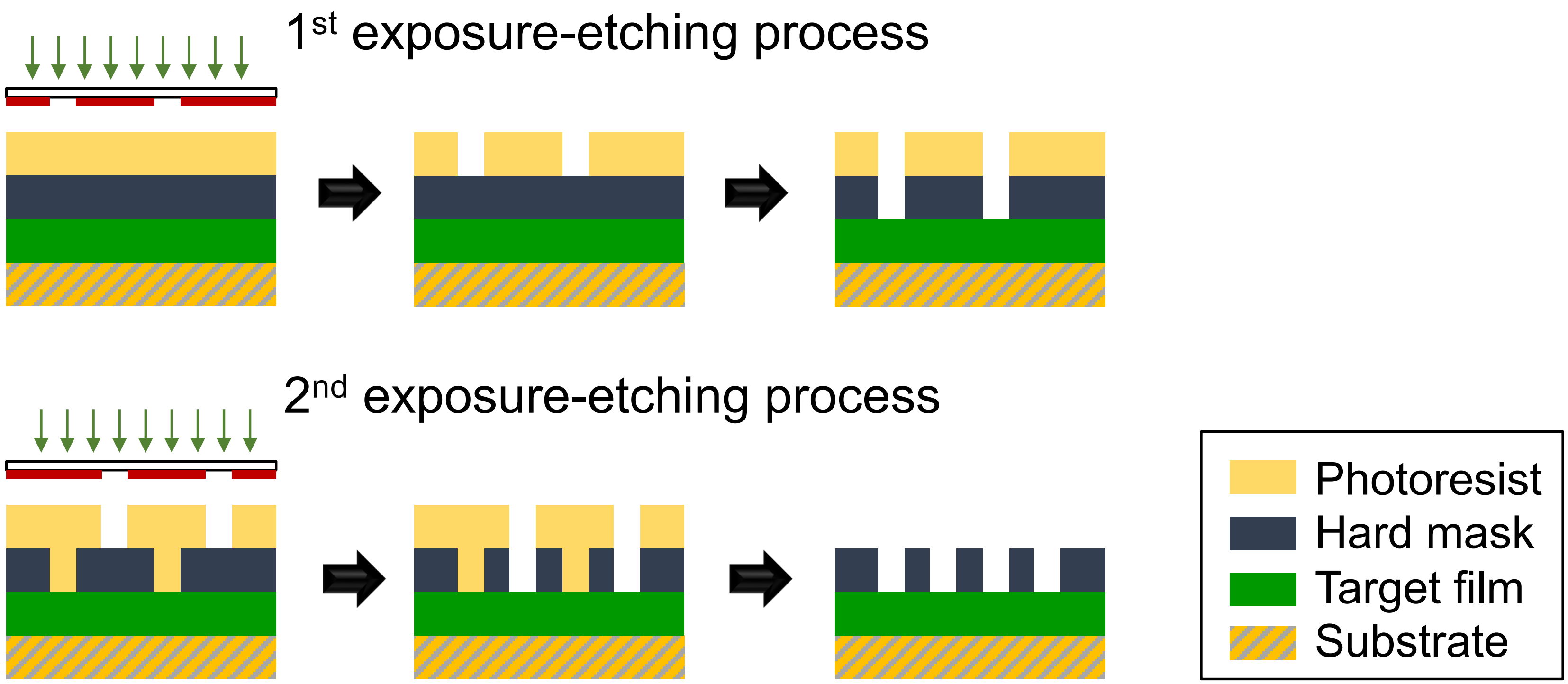
- Decomposes the critical pattern into **two sub-patterns** and then uses two masks to form two sub-patterns
  - Principle: pitch relaxation



critical pattern (advanced technology) ***sub-pattern #1 (mature technology)*** ***sub-pattern #2 (mature technology)***

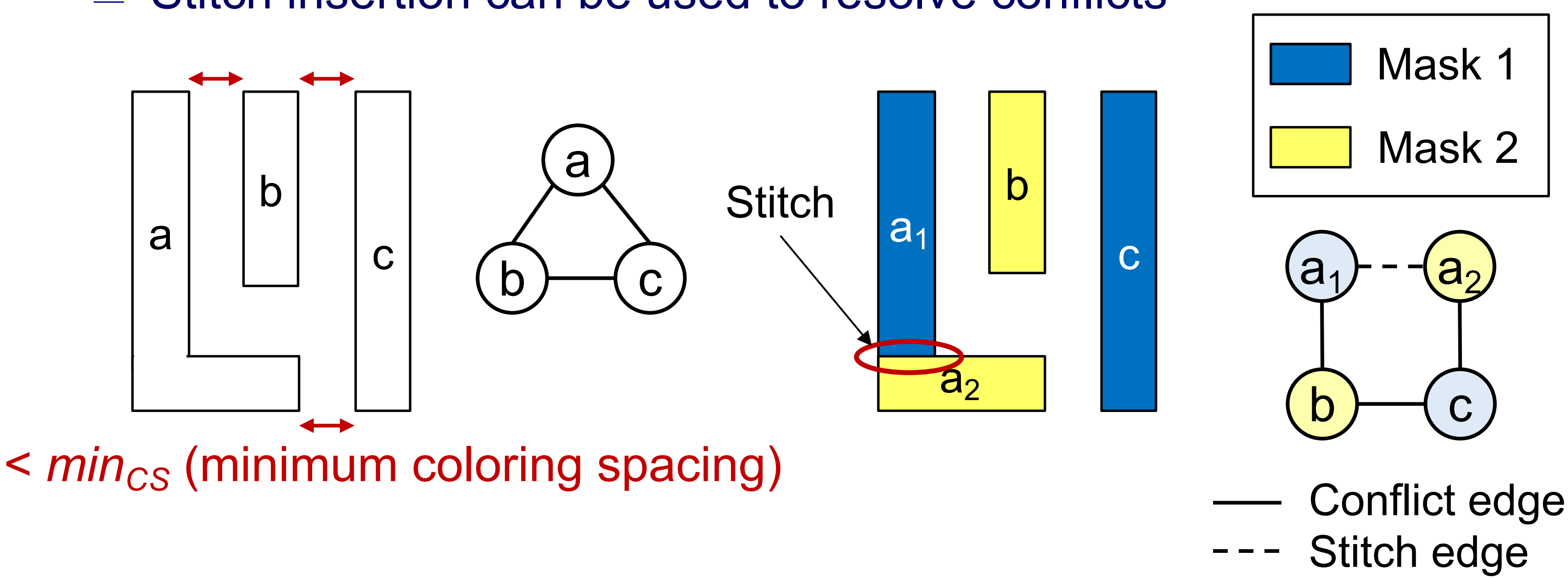
Process of DPL

- Use two times of litho-etching process (thus is called as LELE double patterning)



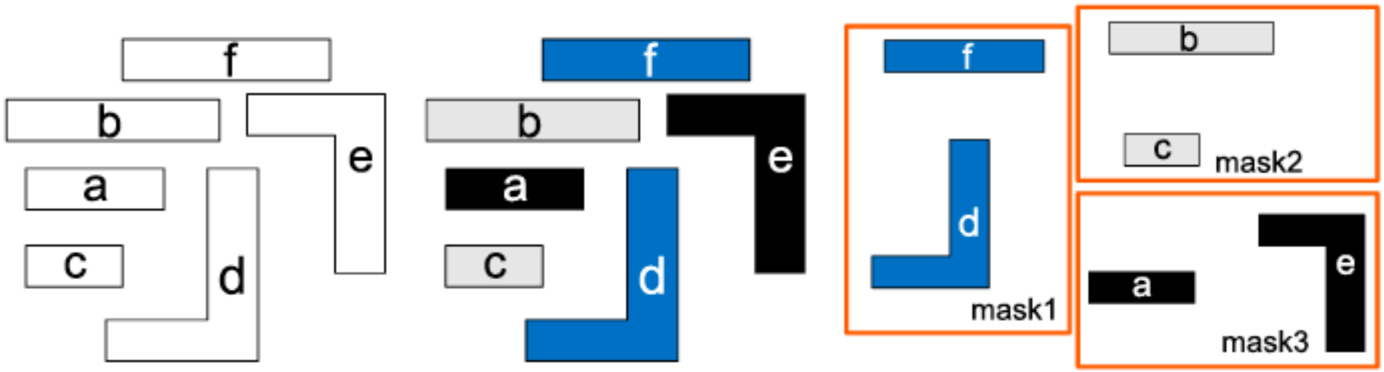
Layout Decomposition (LD)

- ***Layout decomposition:*** assign each pattern to one of the multiple masks while conflicts are minimized
  - Transform an input layout into a conflict graph
  - Two-coloring the conflict graph corresponding to DPL LD
  - Odd cycles are uncolorable with two colors
  - Stitch insertion can be used to resolve conflicts

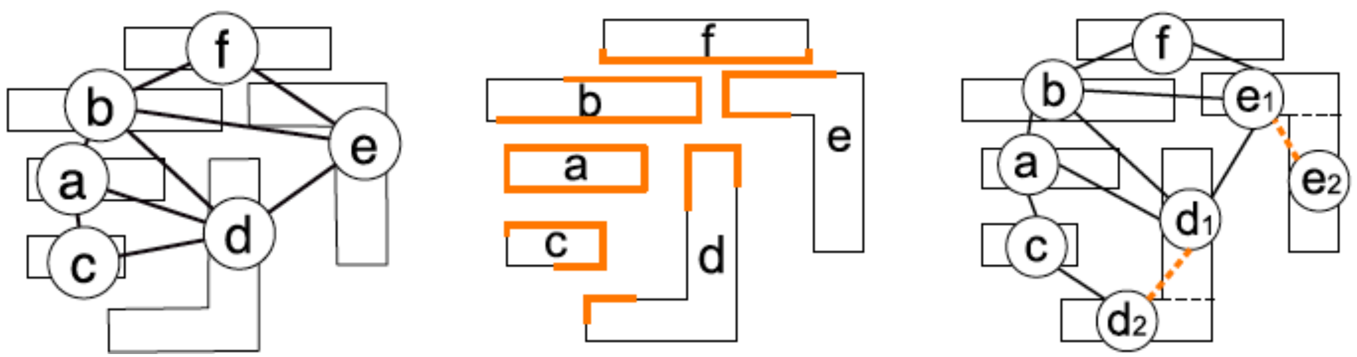


Triple Patterning Lithography (TPL)

- With the advance of process nodes, three masks are required and thus TPL



- Layout decomposition for TPL is basically the same as DPL with higher problem complexity

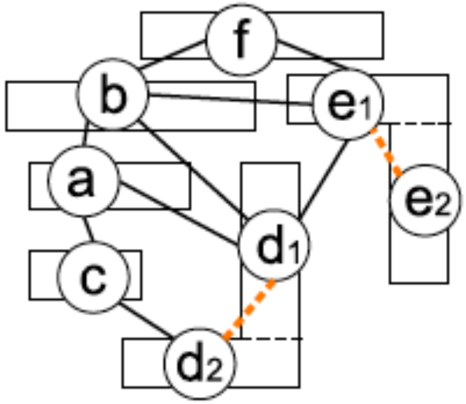


Complexity of Layout Decomposition

- The complexity of graph coloring problem with k masks
  - 2 coloring can be done in polynomial (e.g., with BFS)
  - k coloring when $k > 2$ is NP-complete
  - k coloring when $k \geq 2$ is NP-complete if stitch insertion is used
- The objectives of DPL/TPL LD
  - Minimize #coloring conflicts between each pair of patterns within the minimum coloring spacing
  - Minimize #stitches inserted on patterns
  - Conflicts should have higher priority to be minimized

ILP Notations for DPL

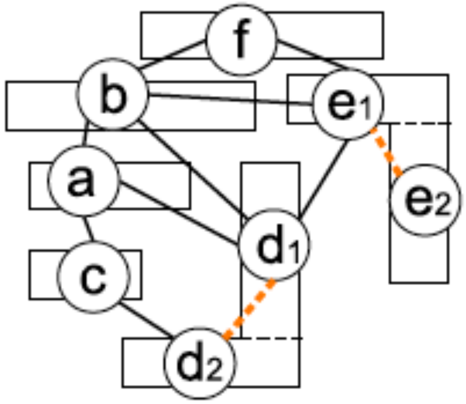
- Notations for DPL
  - CE: a set of conflict edges
  - SE: a set of stitch edges
  - V: a set of vertices representing layout patterns
  - v_i: the i-th vertex
  - e_{ij}: an edge connecting v_i and v_j
  - x_i: binary variable indicating the color of v_i; $x_i = 0/1$ means v_i is assigned to the first/second color (mask)
  - c_{ij}: binary variable; $c_{ij} = 1$ if a coloring conflict exists between v_i and v_j; $c_{ij} = 0$ otherwise
  - s_{ij}: binary variable; $s_{ij} = 1$ if a stitch is inserted between v_i and v_j; $s_{ij} = 0$ otherwise



ILP Formulation for DPL LD

- Objective

$$\sum_{e_{ij}\in CE} c_{ij} + \alpha\times \sum_{e_{ij}\in SE} s_{ij}$$



- Constraints

$$x_i + x_j \leq 1 + c_{ij}, \forall e_{ij} \in CE$$

$$(1 - x_i) + \left(1 - x_j\right) \leq 1 + c_{ij}, \forall e_{ij} \in CE$$

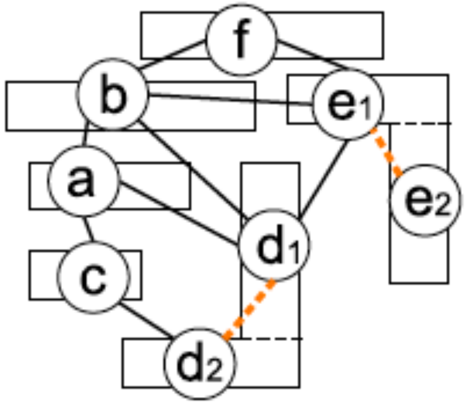
$$x_i - x_j \leq s_{ij}, \forall e_{ij} \in SE$$

$$x_j - x_i \leq s_{ij}, \forall e_{ij} \in SE$$

$$x_i = 0 \text{ or } 1, \forall v_i \in V$$

ILP Notations for TPL

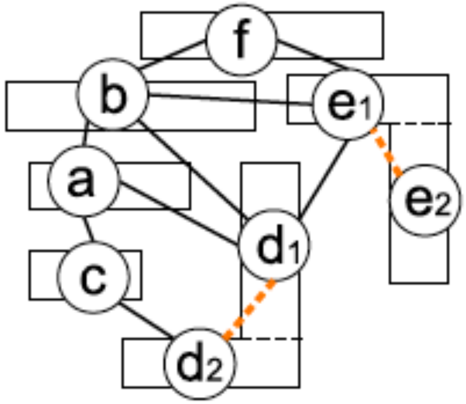
- Notations for TPL
  - CE: a set of conflict edges
  - SE: a set of stitch edges
  - V: a set of vertices representing layout patterns
  - v_i: the i-th vertex
  - e_{ij}: an edge connecting v_i and v_j
  - c_{ij}: binary variable; $c_{ij} = 1$ if a coloring conflict exists between v_i and v_j
  - s_{ij}: binary variable; $s_{ij} = 1$ if a stitch is inserted between v_i and v_j
  - x_{i1}, x_{i2}: binary variable indicating the color of v_i; $(x_{i1}, x_{i2}) = (0,0)/(0,1)$ $/(1,0)$ means v_i is assigned to the first/second/third color (mask)
  - c_{ij1}, c_{ij2}: binary variable; $c_{ijk} = 1$ if $x_{ik} = x_{jk}$; $c_{ijk} = 0$ otherwise
  - s_{ij1}, s_{ij2}: binary variable; $s_{ijk} = 1$ if $x_{ik} \neq x_{jk}$; $s_{ijk} = 0$ otherwise



ILP Formulation for TPL LD

- Objective

$$\sum_{e_{ij} \in CE} c_{ij} + \alpha \times \sum_{e_{ij} \in SE} s_{ij}$$



- Constraint for the three available colors

$$x_{i1} + x_{i2} \leq 1, \forall v_i \in V$$

$$x_{i1}, x_{i2} = 0 \text{ or } 1, \forall v_i \in V$$

- Constraints for conflict detection

$$x_{i1} + x_{j1} \leq 1 + c_{ij1}, \forall e_{ij} \in CE$$

$$(1 - x_{i1}) + \left(1 - x_{j1}\right) \leq 1 + c_{ij1}, \forall e_{ij} \in CE$$

$$x_{i2} + x_{j2} \leq 1 + c_{ij2}, \forall e_{ij} \in CE$$

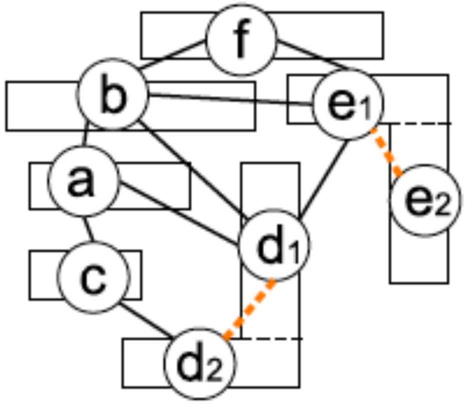
$$(1 - x_{i2}) + \left(1 - x_{j2}\right) \leq 1 + c_{ij2}, \forall e_{ij} \in CE$$

$$c_{ij1} + c_{ij2} \leq 1 + c_{ij} \ , \forall e_{ij} \in CE$$

ILP Formulation for TPL LD (cont'd)

- Objective

$$\sum_{e_{ij} \in CE} c_{ij} + \alpha \times \sum_{e_{ij} \in SE} s_{ij}$$



- Constraints for stitch detection

$$x_{i1} - x_{j1} \leq s_{ij1}, \forall e_{ij} \in SE$$
$$x_{j1} - x_{i1} \leq s_{ij1}, \forall e_{ij} \in SE$$
$$x_{i2} - x_{j2} \leq s_{ij2}, \forall e_{ij} \in SE$$
$$x_{j2} - x_{i2} \leq s_{ij2}, \forall e_{ij} \in SE$$
$$s_{ij} \geq s_{ij1}, \forall e_{ij} \in SE$$
$$s_{ij} \geq s_{ij2}, \forall e_{ij} \in SE$$