

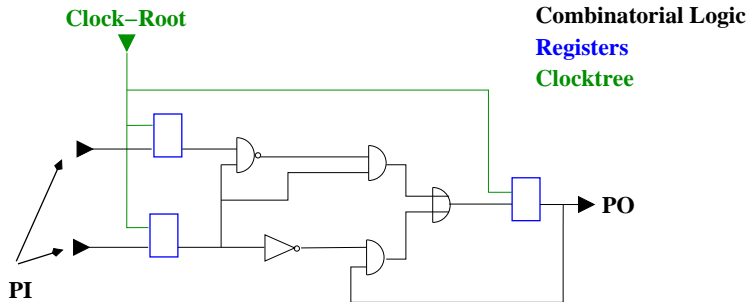
# Gate- and Wire-Sizing in VLSI Design

Dieter Rautenbach

Forschungsinstitut für Diskrete Mathematik  
Universität Bonn

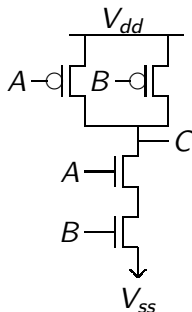
21. Juni 2006

# ASIC=Application Specific Integrated Circuit



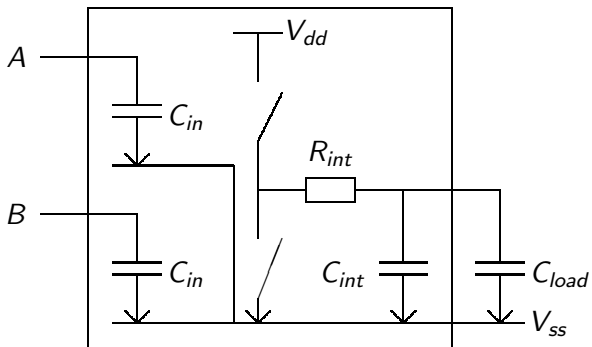
We see gates whose sizes we can actually choose.

# A simple gate



If the input signals change, internal structures are charged / discharged, the transistors isolate / connect and internal structures as well as subsequent structures are charged / discharged.

# A simple electrical model for a gate



# A simple electrical model for a gate, II

The electrical behaviour of a gate  $g$  which has only one output pin can be approximately analyzed using the gate's

- input capacity  $C_{in}(g)$  (or capacities),
- internal (“intrinsic”) resistance  $R_{int}(g)$ ,
- internal (“intrinsic”) capacitance  $C_{int}(g)$  and
- load capacitance  $C_{load}(g)$ .

- Standard libraries typically offer several different gates for the same logic function.
- The main difference of the alternative gates is their “gate size”.
- Further differences might concern delay properties (e.g. balanced delays for rising/falling signals) or the internal structure.
- The choice of the different gate sizes can approximatively be modeled using a sizing factor.

# The sizing factor of a gate

If we size the internal structures (transistors, wires,...) of  $g$  using a sizing factor

$$x \in [l(g), u(g)] \subseteq \mathbb{R}_{\geq 0},$$

then

$$\begin{aligned}C_{in} &= \bar{C}_{in}(g)x, \\C_{int} &= \bar{C}_{int}(g)x \text{ and} \\R_{int} &= \frac{\bar{R}_{int}(g)}{x}.\end{aligned}$$

# The sizing factor of a gate, II

The load capacitance  $C_{load}(g)$  driven by  $g$  is the sum of the wire capacitance

$$C_{wire}(g)$$

driven by  $g$  and the input capacitances

$$\sum_{h \text{ driven by } g} C_{in}(h)$$

of the driven gates. Each of these input capacitances depends - once again - on the individual gate size.

# Gate sizing problem(s)

- The task of gate sizing is to determine individual (and available) sizes for all gates such that...
  - ...the worst slack is maximized subject to area/power constraints.
  - ...the area/power consumption is minimized subject to timing constraints.
  - ...the design is robust and meets the timing/area/power constraints.
- The theoretically most well-founded/appealing approaches to gate sizing rely on convex/geometric programming methods combined with rounding procedures.
- In practice greedy local heuristics that choose gate sizes taking for instance slew- and capacity-limits into account often yield competitive results.

Using the Elmore delay model, the delay through the gate  $g$  is

$$DELAY(g) = R_{int}(g)(C_{int}(g) + C_{load}(g)).$$

If some gate  $g_0$  of size  $x_0$  drives gates  $g_1, g_2, \dots, g_l$  of sizes  $x_1, x_2, \dots, x_l$ , then

$$DELAY(g) = \frac{\bar{R}_{int}(g_0)}{x_0} \left( \bar{C}_{int}(g_0)x_0 + C_{wire}(g_0) + \sum_{i=1}^l \bar{C}_{in}(g_i)x_i \right).$$

Not changing the wiring, the delay through the wiring from the output pin of  $g_0$  to the driven input pin of  $g_i$  is of the form

$$const_1 + const_2 x_i.$$

Altogether, the delay along a path  $P$  in the design is always of the form

$$DEALY(P) = \sum_{i,j} a_{i,j} \frac{x_i}{x_j} + \sum_i \left( b_i x_i + \frac{c_i}{x_i} \right) + d$$

for appropriate gates sizes

$$x = (x_1, \dots, x_n) \in [l, u] = \times_{i=1}^n [l(g_i), u(g_i)] \subseteq \mathbb{R}_{\geq 0}^n.$$

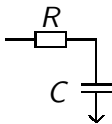
and for appropriate constants

$$a_{i,j}, b_i, c_i, d \in \mathbb{R}_{\geq 0}.$$

The area consumption of all gates can be approximated by

$$\sum_{i=1}^n \alpha_i X_i.$$

# Dynamic power consumption



The charge  $Q$  on the capacitor with capacitance  $C$  relates on the voltage  $V$  via

$$Q = CV.$$

The power consumption for a charging process equals

$$\begin{aligned} \int_{t=0}^{t_{\max}} I(t)V(t)dt &= \int_{t=0}^{t_{\max}} C \frac{dV(t)}{dt} V(t)dt \\ &= \int_{V=V_{\min}}^{V_{\max}} CVdV = C \left( \frac{V_{\max}^2}{2} - \frac{V_{\min}^2}{2} \right). \end{aligned}$$

# Dynamic power consumption, II

Each individual charging/discharging process driven by the gate  $g$  uses the energy

$$(C_{in}(g) + C_{load}(g)) \left( \frac{V_{dd}^2}{2} - \frac{V_{ss}^2}{2} \right).$$

Hence, the overall dynamic power consumption of all gates is also of the form

$$\sum_{i=1}^n \beta_i x_i$$

where the constants  $\beta_i$  are influenced by  $\left( \frac{V_{dd}^2}{2} - \frac{V_{ss}^2}{2} \right)$  as well as the estimated switching frequencies.

The static/leakage power of a gate equals the product of the leakage current and  $(V_{dd} - V_{ss})$ . If we model the leakage current as proportional to the gate size, then also the overall static power consumption of all gates is of the form

$$\sum_{i=1}^n \gamma_i x_i.$$

# A first formulation of gate sizing

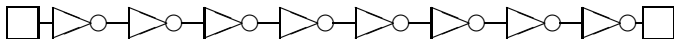
Maximizing the worst slack of a design subject to area- and power-constraints can now be formulated as follows

$$\begin{aligned} \min \quad & \max \{ \text{DELAY}(P) \mid P \text{ relevant path in the design} \} \\ \text{s.th.} \quad & \sum_{i=1}^n \alpha_i x_i \leq \alpha_{\max} \\ & \sum_{i=1}^n (\beta_i + \gamma_i) x_i \leq \beta_{\max} + \gamma_{\max} \\ & x \in [l, u]. \end{aligned}$$

Note that  $\text{DELAY}(P)$  and the functions describing the constraints are so-called “posynomials”.

# A simple case

Consider a chain  $C$  of  $N$  identical inverters.



Ignoring the wiring, the load capacitance of one inverter is the sum of its own intrinsic capacitance and the input capacitance of the subsequent inverter. Thus the delay along this chain is of the form

$$DELAY(C) = const_1 + const_2 \sum_{i=1}^N \frac{x_{i+1}}{x_i} + const_3 \frac{1}{x_N}.$$

Setting

$$\frac{\partial}{\partial x_i} DELAY(C) = 0$$

yields

$$-\frac{x_{i+1}}{x_i^2} + \frac{1}{x_{i-1}} = 0,$$

i.e.

$$\frac{x_i}{x_{i-1}} = \frac{x_{i+1}}{x_i}.$$

$\Rightarrow$

Geometrically increasing gate sizes would be optimal and we could write down a closed formula for the optimal delay.

D. Harris et al.: “Designing for speed on the back of an envelope...”. The delay  $d$  through a gate is estimated as

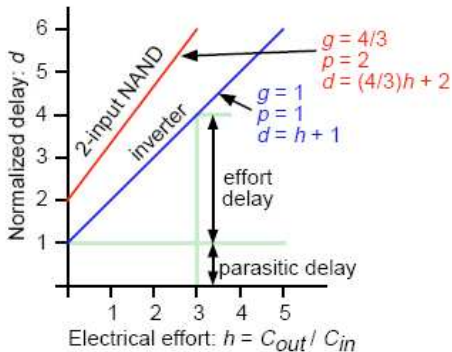
$$d = gh + p = g \frac{C_{out}}{C_{in}} + p$$

where

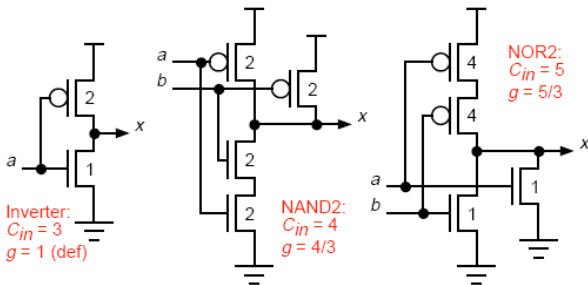
- $d$  is measured in technology-independent units,
- $p$  is a “parasitic” gate delay,
- $h = \frac{C_{out}}{C_{in}}$  is the “electrical effort” and
- $g$  is the “logical effort” that captures the internal gate structure.

# Logical Effort, II

The equation  $d = gh + p$  is justified, since the delay plots look approximately like this

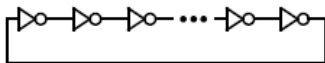


The logical effort  $g$  of some gate depends on its internal structure.



# Typical calculations using Logical Effort

The delay of an oscillator chain of  $N$  identical inverters.



Logical Effort:  $g \equiv 1$

Electrical Effort:  $h = \frac{C_{out}}{C_{in}} = 1$

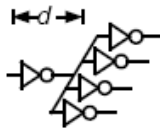
Parasitic Delay:  $p = p_{inv} \approx 1$

Stage Delay:  $d = gh + p = 2$

Oscillator Frequency:  $F = \frac{1}{2Nd_{abs}} = \frac{1}{4N\tau}$

# Typical calculations using Logical Effort, II

The delay of one inverter driving four identical inverters.



Logical Effort:  $g \equiv 1$

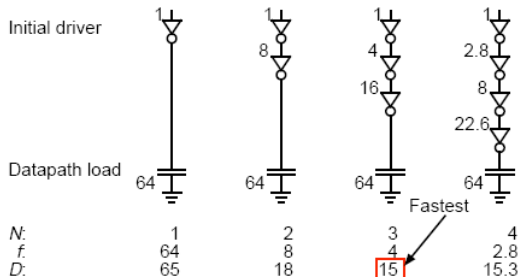
Electrical Effort:  $h = \frac{C_{out}}{C_{in}} = 4$

Parasitic Delay:  $p = p_{inv} \approx 1$

Stage Delay:  $d = gh + p = 5$

# Typical calculations using Logical Effort, III

The best number of stages in an inverter chain driving a certain capacitance.



- Logical Effort is a collection of “rules of thumb” that can be derived from the convex programming formulations.
- For instance “...the delay is minimized when each stage bears the same effort...”.
- It neglects slew- and interconnect-effects.
- It is intended for speed optimization only and does not allow to minimize area/power consumption subject to timing constraints.

⇒ Back to the above formulation.

# (Generalized) geometric programming

Given positive reals  $x_1, x_2, \dots, x_n$ , a monomial is a term of

$$c \prod_{i=1}^n x_i^{\alpha_i}$$

for  $c > 0$  and  $\alpha_1, \alpha_2, \dots, \alpha_n \in \mathbb{R}$ . A posynomial is the sum of monomials. A generalized posynomial is the pointwise maximum of posynomials. A (generalized) geometric program is an optimization problem of the form

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 1 \\ & g_j(x) = 1 \end{aligned}$$

where  $f_0$  and the  $f_i$  are (generalized) posynomials and the  $g_j$  are monomials.

# The convexity of geometric programs

Let  $0 \leq \theta \leq 1$  and  $x, \tilde{x} \in \mathbb{R}_{>0}^n$ . Note that we have

$$\begin{aligned} c \prod_{i=1}^n \left( x_i^\theta \tilde{x}_i^{(1-\theta)} \right)^{\alpha_i} &= c \prod_{i=1}^n x_i^{\alpha_i \theta} \prod_{i=1}^n \tilde{x}_i^{\alpha_i (1-\theta)} \\ &= \left( c \prod_{i=1}^n x_i^{\alpha_i} \right)^\theta \left( c \prod_{i=1}^n \tilde{x}_i^{\alpha_i} \right)^{(1-\theta)}. \end{aligned}$$

Note furthermore that by Hölder's inequality ( $\Leftarrow$  concavity of the logarithm)

$$\begin{aligned} & \left( \sum_{i=1}^n x_i \right)^\theta \left( \sum_{i=1}^n \tilde{x}_i \right)^{(1-\theta)} \\ = & \left( \sum_{i=1}^n \left( x_i^\theta \right)^{\frac{1}{\theta}} \right)^\theta \left( \sum_{i=1}^n \left( \tilde{x}_i^{(1-\theta)} \right)^{\frac{1}{(1-\theta)}} \right)^{(1-\theta)} \\ \geq & \sum_{i=1}^n \left( x_i^\theta \tilde{x}_i^{(1-\theta)} \right). \end{aligned}$$

# The convexity of geometric programs, III

Putting these two observations together, we obtain for a posynomial  $f$  that

$$\begin{aligned} & f\left(\left(x_1^\theta \tilde{x}_1^{(1-\theta)}, \dots, x_n^\theta \tilde{x}_n^{(1-\theta)}\right)\right) \\ &= \sum_{g \text{ monomial}} g\left(\left(x_1^\theta \tilde{x}_1^{(1-\theta)}, \dots, x_n^\theta \tilde{x}_n^{(1-\theta)}\right)\right) \\ &= \sum_{g \text{ monomial}} \left(g(x)^\theta g(\tilde{x})^{(1-\theta)}\right) \\ &\leq \left(\sum_{g \text{ monomial}} g(x)\right)^\theta \left(\sum_{g \text{ monomial}} g(\tilde{x})\right)^{(1-\theta)} \\ &= f(x)^\theta f(\tilde{x})^{(1-\theta)}. \end{aligned}$$

# The convexity of geometric programs, IV

If we introduce new variables  $y_i$  with

$$x_i = e^{y_i},$$

then the above implies that the function

$$F(y) = \log (f (e^{y_1}, e^{y_2}, \dots, e^{y_n}))$$

is convex.

$\Rightarrow$

The given formulation of the gate sizing problem can be transformed into a convex programming problem.

# Timing constraints

If we introduce signal arrival times  $AT(u)$  for all relevant nodes (pins) of the timing graph  $G = (V, E)$ , then we can obtain the following formulation

$$\begin{aligned} \min \quad & AT_{\max} \\ \text{s.th.} \quad & 0 \leq AT(u) \leq AT_{\max} \quad \forall u \in V \\ & AT(u) + DELAY((u, v)) \leq AT(v) \quad \forall (u, v) \in E \\ & \sum_{i=1}^n \alpha_i x_i \leq \alpha_{\max}, \quad \dots \quad x \in [l, u]. \end{aligned}$$

Note that  $DELAY((u, v))$  is once again a posynomial depending “sparsly” on the gate sizes  $x$ .

- Interior point methods can solve such problems within a few second for designs with  $10^3$  gates. (Boyd, Kim et al. 2004).
- Several further constraints can be incorporated within this framework.
- In 1998 Chen, Chu and Wong proposed a different approach based on Lagrange duality and the subgradient projection method.

# Dualizing the timing constraints

Consider the following version of the gate sizing problem

$$\begin{aligned} \min \quad & \sum_{i=1}^n \alpha_i x_i \\ \text{s.t.} \quad & AT(u) + DELAY((u, v)) \leq AT(v) \quad \forall (u, v) \in E \\ & x \in [l, u] \text{ and some arrival times } AT(u) \text{ fixed.} \end{aligned}$$

The fixed arrival times correspond to given arrival times at the primary input pins or latch output pins and to the required arrival times at primary output pins or latch input pins. The timing constraints reflect worst case late mode timing.

# Dualizing the timing constraints, II

If we dualize the timing constraints using Lagrange multipliers  $\lambda = (\lambda_{(u,v)})_{(u,v) \in E} \in \mathbb{R}_{\geq 0}^E$ , then we obtain the following Lagrange function

$$\begin{aligned} & L(x, AT, \lambda) \\ = & \sum_{i=1}^n \alpha_i x_i + \sum_{(u,v) \in E} \lambda_{(u,v)} (AT(u) + DELAY((u, v)) - AT(v)) \\ = & \left( \sum_{i=1}^n \alpha_i x_i + \sum_{(u,v) \in E} \lambda_{(u,v)} DELAY((u, v)) \right) \\ & + \left( \sum_{(u,v) \in E} \lambda_{(u,v)} (AT(u) - AT(v)) \right) \\ = & L_1(x, \lambda) + L_2(AT, \lambda). \end{aligned}$$

Note that

$$\begin{aligned} L_2(AT, \lambda) &= \sum_{(u,v) \in E} \lambda_{(u,v)} (AT(u) - AT(v)) \\ &= \sum_{u \in V} AT(u) \left( \sum_{v: (u,v) \in E} \lambda_{(u,v)} - \sum_{v: (v,u) \in E} \lambda_{(v,u)} \right). \end{aligned}$$

Note furthermore that after dualizing the timing constraints most arrival times are free variables.

By Lagrange duality, we have that the optimum value of the considered problem equals

$$\begin{aligned} \max \quad & D(\lambda) \\ \text{s.th.} \quad & \lambda \geq 0. \end{aligned}$$

where  $D(\lambda)$  equals

$$\begin{aligned} \min \quad & L(x, AT, \lambda) \\ \text{s.th.} \quad & x \in [l, u] \text{ and some arrival times } AT(u) \text{ fixed.} \end{aligned}$$

# The multipliers form a flow

Since

$$L_2(AT, \lambda) = \sum_{u \in V} AT(u) \left( \sum_{v: (u,v) \in E} \lambda_{(u,v)} - \sum_{v: (v,u) \in E} \lambda_{(v,u)} \right),$$

the last problem is unbounded whenever the multipliers  $\lambda_{(u,v)}$  do not satisfy the flow condition

$$\sum_{v: (u,v) \in E} \lambda_{(u,v)} = \sum_{v: (v,u) \in E} \lambda_{(v,u)}$$

for all vertices  $u \in V$  for which  $AT(u)$  is not one of the fixed values. Therefore, we can restrict ourselves to multiplier vectors  $\lambda$  that define a non-negative flow vector on the timing graph.

# The simplified dual problem

If  $\lambda$  defines a non-negative flow, then  $L_2(AT, \lambda)$  is constant as a function of  $AT$  and the dual problem reduces to

$$\begin{aligned} \max \quad & D(\lambda) \\ \text{s.th.} \quad & \lambda \text{ defines a non-negative flow.} \end{aligned}$$

where  $D(\lambda)$  equals

$$\begin{aligned} \min \quad & L_1(x, \lambda) \\ \text{s.th.} \quad & x \in [l, u]. \end{aligned}$$

Solving the last problem is usually called “local refinement”.

We consider the minimization problem

$$\min\{f(x) \mid x \in [l, u]\} \quad (1)$$

for a function  $f : \mathbb{R}_{>0}^n \rightarrow \mathbb{R}_{>0}$  defined by

$$f(x) = f((x_1, \dots, x_n)) = \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \frac{x_i}{x_j} + \sum_{i=1}^n \left( b_i x_i + \frac{c_i}{x_i} \right)$$

for appropriate constants  $a_{i,j}, b_i, c_i$  over an  $n$ -dimensional interval

$$[l, u] = [l_1, u_1] \times \dots \times [l_n, u_n].$$

Saddle point optimality conditions from convex optimization easily imply the following

### Lemma

*A vector  $x \in [l, u]$  is an optimum solution of (1) if and only if the following conditions hold for  $1 \leq i \leq n$ :*

$$\frac{\partial}{\partial x_i} f(x) < 0 \Rightarrow x_i = u_i$$

*and*

$$\frac{\partial}{\partial x_i} f(x) > 0 \Rightarrow x_i = l_i.$$

For  $1 \leq i \leq n$  we have

$$\begin{aligned}
 \frac{\partial}{\partial x_i} f(x) &= \frac{\partial}{\partial x_i} \left( \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \frac{x_i}{x_j} + \sum_{i=1}^n \left( b_i x_i + \frac{c_i}{x_i} \right) \right) \\
 &= \sum_{j: 1 \leq j \leq n, j \neq i} \left( \frac{a_{i,j}}{x_j} - \frac{a_{j,i} x_j}{x_i^2} \right) + b_i - \frac{c_i}{x_i^2} \\
 &= \left( b_i + \sum_{j: 1 \leq j \leq n, j \neq i} \frac{a_{i,j}}{x_j} \right) - \left( c_i + \sum_{j: 1 \leq j \leq n, j \neq i} a_{j,i} x_j \right) \frac{1}{x_i^2} \\
 &= (b_i + B_i(x)) - \frac{(c_i + C_i(x))}{x_i^2}.
 \end{aligned}$$

Note that  $B_i(x)$  and  $C_i(x)$  are both independent of  $x_i$ .

If we set

$$T_i(x) = \sqrt{\frac{c_i + C_i(x)}{b_i + B_i(x)}},$$

then

$$\frac{\partial}{\partial x_i} f(x) = 0 \Leftrightarrow x_i = T_i(x),$$

$$\frac{\partial}{\partial x_i} f(x) > 0 \Leftrightarrow x_i > T_i(x)$$

and

$$\frac{\partial}{\partial x_i} f(x) < 0 \Leftrightarrow x_i < T_i(x).$$

If we set

$$\bar{T}_i(x) = \max\{l_i, \min\{u_i, T_i(x)\}\}$$

and

$$\bar{T}(x) = (\bar{T}_1(x), \dots, \bar{T}_n(x)),$$

then  $\bar{T}_i(x)$  is the optimum value of the  $i$ -th coordinate fixing all other coordinates and the above lemma reads as follows.

## Lemma

*A vector  $x \in [l, u]$  is an optimum solution of (1) if and only if  $x = \bar{T}(x)$ .*

For

$$\Delta = \max \left\{ \frac{u_i - l_i}{l_i} \mid 1 \leq i \leq n \right\}$$

we obtain the following

## Theorem

If  $x^1 \in [l, u]$  and  $x^{k+1} = \overline{T}(x^k)$  for  $k \in \mathbb{N}$ , then  $(x^k)_{k \in \mathbb{N}}$  converges to an optimum solution  $\bar{x}$  of (1) and

$$\left| \frac{x_i^k - \bar{x}_i}{\bar{x}_i} \right| \leq \frac{\Delta(\Delta + 1)\rho^k}{1 - \rho}$$

for  $1 \leq i \leq n$ .

i.e. the method of cyclic relaxation converges linearly to an optimum solution when applied to (1).

## A subgradient for $-D$

Having solved the 'local refinement' for some  $\lambda$ , we can easily determine a subgradient for  $-D(\lambda)$ .

$$\begin{aligned}D(\tilde{\lambda}) &= \min\{L_1(x, \tilde{\lambda}) \mid x \in [l, u]\} \\ &\leq L_1(x(\lambda), \tilde{\lambda}) \\ &= \alpha^T x(\lambda) + \tilde{\lambda}^T \text{DELAY}(x(\lambda)) \\ &= \alpha^T x(\lambda) + \lambda^T \text{DELAY}(x(\lambda)) + (\tilde{\lambda} - \lambda)^T \text{DELAY}(x(\lambda)) \\ &= D(\lambda) + (\tilde{\lambda} - \lambda)^T \text{DELAY}(x(\lambda)),\end{aligned}$$

i.e. the vector  $\text{DELAY}(x(\lambda))$  is a subgradient at  $\lambda$ .

# Constrained Subgradient Method (Ermoliev and Polyak)

Given a convex function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ , a closed, convex domain  $X \subseteq \mathbb{R}^n$ , a sequence of multipliers  $(\rho_k)_{k \in \mathbb{N}}$  in  $\mathbb{R}$  and a starting point  $x_1 \in \mathbb{R}^n$  the method produces the sequence  $(x_k)_{k \in \mathbb{N}}$  of points in  $\mathbb{R}^n$  with

$$x_{k+1} = \pi_X(x_k - \rho_k n(\gamma_k)) \quad (2)$$

for  $k \geq 1$  where  $\gamma_k$  denotes a subgradient of  $f$  in  $x_k$ , i.e.

$$f(x) \geq f(x_k) + \gamma_k^T(x - x_k) \quad \forall x \in \mathbb{R}^n,$$

$$n(x) = \begin{cases} \frac{x}{\|x\|} & , x \neq 0 \\ 0 & , x = 0 \end{cases}$$

and  $\pi_X$  denotes the (well-defined and unique) projection onto  $X$ , i.e.  $\pi_X(x) = \arg \min_{y \in X} \|y - x\|$ .

# Constrained Subgradient Method (Ermoliev and Polyak), II

The basic convergence result for this method is the following theorem.

## Theorem (Polyak,...)

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be convex and  $X \subseteq \mathbb{R}^n$  be closed and convex such that either

$$\lim_{\|x\| \rightarrow \infty} f(x) = \infty$$

or  $X$  is compact.

If  $(x_k)_{k \in \mathbb{N}}$  is defined as in (2) for  $x_1 \in \mathbb{R}^n$  and  $(\rho_k)_{k \in \mathbb{N}}$  with  $\rho_k > 0$  for  $k \in \mathbb{N}$  and

$$\sum_{k=1}^{\infty} \rho_k = \infty,$$

then

$$\liminf_{k \rightarrow \infty} f(x_k) = \min_{x \in X} f(x).$$

- Applied to the considered gate sizing problem the most time consuming step would be the projection of the  $\lambda$  vectors onto the space of non-negative flows on the timing graph.
- No previous implementation did this projection exactly (Chen and Wong, Muuss,...).
- But, if the projection is not carried out exactly, all convergence guarantees are void.

The following result is well-known.

## Theorem (Cheney and Goldstein)

Let  $\mathcal{C} = \{C_1, C_2, \dots, C_m\}$  be a collection of closed, convex sets in  $\mathbb{R}^n$  with non-empty intersection  $X$ .

If  $x_1 \in \mathbb{R}^n$  and

$$x_{k+1} = \pi_{C_m} \circ \pi_{C_{m-1}} \circ \dots \circ \pi_{C_1}(x_k)$$

for  $k \geq 1$ , then  $(x_k)_{k \in \mathbb{N}}$  converges to a point in  $X$ .

In view of this result, we propose the following version of the Constrained Subgradient Method.

# Constrained Subgradient Method using Alternating Projections

Given a convex function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ , a collection  $\mathcal{C} = \{C_1, C_2, \dots, C_m\}$  of closed, convex sets in  $\mathbb{R}^n$  with non-empty intersection  $X$ , a sequence of multipliers  $(\rho_k)_{k \in \mathbb{N}}$  in  $\mathbb{R}$  and a starting point  $x_1 \in \mathbb{R}^n$  we define

$$x_{k+1} = \pi_{C_m} \circ \pi_{C_{m-1}} \circ \dots \circ \pi_{C_1} (x_k - \rho_k n(\gamma_k)) \quad (3)$$

for  $k \geq 1$  where  $\gamma_k$  denotes a subgradient of  $f$  in  $x_k$ .

## Definition

*A collection  $\mathcal{C}$  of sets in  $\mathbb{R}^n$  with non-empty intersection  $X$  is said to intersect nicely, if for every  $\epsilon > 0$  there is a  $\delta = \delta(\epsilon, \mathcal{C}) > 0$  such that  $d(x, C) < \delta$  for all  $C \in \mathcal{C}$  and some  $x \in \mathbb{R}^n$  implies  $d(x, X) < \epsilon$ .*

## Theorem

Let  $\mathcal{C} = \{C_1, C_2, \dots, C_m\}$  be a nicely intersecting collection of closed, convex sets in  $\mathbb{R}^n$  with intersection  $X$  and let  $(y_k)_{k \in \mathbb{N}}$  be a sequence of vectors in  $\mathbb{R}^n$  with  $\lim_{k \rightarrow \infty} \|y_k\| = 0$ . Let  $x_1 \in \mathbb{R}^n$  and let

$$x_{k+1} = \pi_{C_m} \circ \pi_{C_{m-1}} \circ \dots \circ \pi_{C_1}(x_k + y_k)$$

for  $k \in \mathbb{N}$ .

For every  $\epsilon > 0$  there is some  $\beta = \beta(\epsilon, \mathcal{C}) > 0$  such that

$$\lim_{k \rightarrow \infty} d(x_k, X) = 0$$

whenever  $d(x_1, X) \leq \epsilon$  and  $\|y_k\| \leq \beta$  for all  $k \in \mathbb{N}$ .

## Theorem

Let  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  be convex and let  $\mathcal{C} = \{C_1, C_2, \dots, C_m\}$  be a collection of closed, convex sets in  $\mathbb{R}^n$  with non-empty intersection  $X$  such that either  $\lim_{\|x\| \rightarrow \infty} f(x) = \infty$  or  $C_m$  is compact. Let  $(x_k)_{k \in \mathbb{N}}$  be defined as in (3) for  $x_1 \in \mathbb{R}^n$  and  $(\rho_k)_{k \in \mathbb{N}}$  with  $\rho_k > 0$  for  $k \in \mathbb{N}$  and  $\sum_{k=1}^{\infty} \rho_k = \infty$ .

If either  $\mathcal{C}$  is nicely intersecting or  $(x_k)_{k \in \mathbb{N}}$  is bounded, then

$$\liminf_{k \rightarrow \infty} f(x_k) = \min_{y \in X} f(y).$$

Furthermore, a subsequence of  $(x_k)_{k \in \mathbb{N}}$  converges to an optimum solution of the problem  $\min_{y \in X} f(y)$ .

- The last result allows a stable and fast implementation of the subgradient projection method for the gate sizing problem.
- How do these analytical approaches work in practice? How fast do they converge?
- Can the obtained sizes be realized?
- Are the delay assumptions realistic? What about the slews?
- Are the circuit models realistic? What about compound gates?

# A practical approach - fast and good

We will now show results for the following very simple local and greedy approach.

- First choose minimum gate sizes such that there are no slew- or capacitance-violations.
- Secondly, increase gate sizes on critical paths to improve timing - essentially by trying out different gate sizes.

In order to validate the quality of the obtained gate sizing we determine an upper bound on the achievable slack as follows.

- Perform a timing with the given gate sizes.
- Collect the most critical paths of the design.
- For each critical path do the following:
  - Resize all gates that are driven by gates on the path to the minimum possible size subject to slew- and capacitance-limits.
  - Optimize the gate sizes on the critical path e.g. by complete enumeration.
  - Determine the timing along the path and calculate the slack at its endpoint.
- The minimum over all these slacks is the upper bound to which we compare the achieved slack.

## A practical approach - fast and good, III

Chip	Ckts ( $10^3$ )	Final slack	Final bound	Delay	Ratio
Chip 1	731	-1.593	-1.593	3.666	1.000
Chip 2	2819	-2.948	-2.804	1.512	1.105
Chip 3	71	-0.647	-0.487	4.652	1.036
Chip 4	855	-48.822	-48.697	1.238	1.112
Chip 5	986	-0.850	-0.850	2.961	1.000
Chip 6	1398	-2.028	-1.984	4.582	1.010
Chip 7	302	-0.515	-0.515	1.017	1.000
Chip 8	1104	-0.811	-0.811	0.325	1.000
Chip 9	1050	-0.621	-0.541	3.179	1.026
Chip 10	167	-1.194	-1.190	3.073	1.001
Chip 11	482	-0.929	0.847	5.106	1.016
Chip 12	601	-1.092	-1.029	3.133	1.021
Chip 13	1632	-1.208	-1.073	3.627	1.039

Thank you for your attention!!