Efficient Optimization by Modifying the Objective Function: Applications to Timing-Driven VLSI Layout

Ross Baldick, *Member, IEEE*, Andrew B. Kahng, Andrew Kennings, and Igor L. Markov

***Abstract*—When minimizing a given objective function is challenging because of, for example, combinatorial complexity or points of nondifferentiability, one can apply more efficient and easier-to-implement algorithms to modified versions of the function. In the ideal case, one can employ known algorithms for the modified function that have a thorough theoretical and empirical record and for which public implementations are available. The main requirement here is that minimizers of the objective function not change much through the modification, i.e., the modification must have a bounded effect on the quality of the solution. Review of classic and recent placement algorithms suggests a dichotomy between approaches that either: (a) heuristically minimize a potentially irrelevant objective function (e.g., VLSI placement with quadratic wirelength) motivated by the simplicity and speed of a standard minimization algorithm; or (b) devise elaborate problem-specific minimization heuristics for more relevant objective functions (e.g., VLSI placement with linear wirelength). Smoothness and convexity of the objective functions typically enable efficient minimization. If either characteristic is not present in the objective function, one can modify and/or restrict the objective to special values of parameters to provide the missing properties. After the minimizers of the modified function are found, they can be further improved with respect to the original function by fast local search using only function evaluations. Thus, it is the modification step that deserves most attention. In this paper, we approximate convex nonsmooth continuous functions by convex differentiable functions which are parameterized by a scalar $\beta > 0$ and have convenient limit behavior as $\beta \to 0$. This allows the use of Newton-type algorithms for minimization and, for standard numerical methods, translates into a tradeoff between solution quality and speed. We prove that our methods apply to arbitrary multivariate convex piecewise-linear functions that are widely used in synthesis and analysis of electrical networks [19], [27]. The utility of our approximations is particularly demonstrated for wirelength and nonlinear delay estimations used by analytical placers for VLSI layout, where they lead to more "solvable" problems than those resulting from earlier comparable approaches [29]. For a particular delay estimate, we show that, while convexity is not straightforward to prove, it holds for a certain range of parameters, which, luckily, are representative of "real-world" technologies.**

***Index Terms*—Analytical, approximation, convex, delay, half perimeter, large scale, linear, nonlinear, timing driven, VLSI placement, wirelength.**

Manuscript received August 26, 1999; revised January 21, 2001. The work of R. Baldick was supported by the Texas Advanced Research (Technology) Program under Grant 003658-235. The work of A. B. Kahng was supported by the Cadence Design Systems, Inc. and by the MARCO Gigascale Silicon Research Center. This work was presented in part at the *Asia and South Pacific Design Automation Conference*, Hong Kong, January 1999. This paper was recommended by Associate Editor R. Sridhar.

R. Baldick is with the Electric and Computer Engineering Department, University of Texas, Austin, TX 78712-1084 USA (e-mail: baldick@ece.utexas.edu).

A. B. Kahng is with the Computer Science and Electric and Computer Engineering Deptartments, University of California at San Diego, La Jolla, CA 92093-0114 USA (e-mail: abk@cs.ucsd.edu).

A. Kennings is with Cypress Semiconductor, San Jose, CA 95134 USA (e-mail: ank@cypress.com).

I. L. Markov is with the Electrical Engineering and Computer Science Department, University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: imarkov@eecs.umich.edu).

Publisher Item Identifier S 1057-7122(01)06087-1.

I. Introduction

We consider minimization of convex and continuous objective functions that are differentiable almost everywhere, except for points where directional derivatives disagree, e.g., functions involving absolute values. Examples in VLSI analytic placement include wirelength [1], [28], and delay [13], [14], [29], [27], both of which depend on the absolute value of cell-to-cell distances. Examples abound in other applications, e.g., multifacility location [6], [11], [20] and de-noising in image processing.

Classic-minimization algorithms, e.g., Newton methods and variants [23], [16], [35], assume differentiability and are inapplicable if optima occur at or near points of nondifferentiability.[1] Therefore, problem-specific algorithms have been developed. In several works [1], [11], [20], nondifferentiability has been addressed by function *regularization*, i.e., removing nondifferentiabilities without significantly changing the set of minimizers. The main benefit of a successful regularization is that Newton-type methods become applicable. Their speed improves as the magnitude of the regularization increases, but optima of the regularized objective diverge from those of the original problem. To gauge the tradeoff between the speed and solution quality, our regularization is parameterized by a scalar $\beta \geq 0$, with $\beta = 0$ corresponding to the original function and any $\beta > 0$, giving a smooth function amenable to numerical methods. Reasonable convergence properties as $\beta \to 0$ and problem-independent scaling of β allow the use of the regularized objective, instead of the original objective, for practical applications.

[1]There are more sophisticated methods that can treat nondifferentiability directly. However, they typically complicate algorithms, increase computational effort, and have convergence problems. Examples include subgradient optimization [15], use of an auxiliary variable, and auxiliary inequality constraints ([12], Sec. 4.2.3) or solution of a sequence of problems with updated weights in the objective function ([12], Sec.4.2.3).

This paper proposes new generalized approaches to construct regularizations for given objectives including arbitrary convex piecewise-linear functions that are widely used in synthesis and analysis of electrical networks [19]. The described methods extend those applied to analytical placement in VLSI layout in [1], where the regularized objective lead to a new interpretation of the well-known heuristic GORDIAN-L [28] and suggested a faster algorithm. Comparable regularizations of linear wirelength and path-delay based objectives cannot be produced by previous approaches.[2] Combining our proposed regularization with a novel strictly convex estimate for path-based delay yields problems that are amenable to Newton-type methods, yet are smaller and easier to solve than those produced by [13], [14], [29]. We thus achieve a new outlook on performance-driven analytical placement.

Section II reviews previous work on timing-driven VLSI placement and defines an application domain of interest. Section III describes our proposed methods of function regularization, along with asymptotic analysis, typical examples and a theorem on applicability to arbitrary convex piecewise-linear functions. Applications to timing-driven VLSI placement are given in Section IV followed by empirical validation in Section V, and conclusions in Section VI.

II. Past Work and Motivating Examples

A. Wirelength Approximation

Analytical placers locate cells to minimize wirelength by solving a sequence of optimization problems. Since the exact wiring of nets is unknown until routing occurs, wirelength estimations are used. For instance, one can solve the problem

$$\min_{\mathbf{x}} \left\{ \sum_{i>j} a_{ij}|x_i - x_j| : \mathbf{Hx} = \mathbf{b} \right\} \qquad (1)$$

where $\mathbf{x}$ is the *cell location vector* and each nonnegative number a_{ij} weights the importance of keeping cells i and j in close proximity (when cells i and j are not connected, $a_{ij} = 0$). The matrix $\mathbf{H}$ represents generic *linear* placement constraints via the linear system of equations $\mathbf{Hx} = \mathbf{b}$. Linear constraints include fixed and aligned cells as well as center-of-gravity constraints [28] intended to spread cells evenly throughout the placement region. A similar optimization can be formulated in the y direction. Unfortunately, the presence of $|\cdot|$ implies that the problem, as formulated, is not directly amenable to Newton-type methods since the objective function is neither differentiable nor strictly convex. Linear programming representations of the above problem [33] with 10^4–10^6 unknowns and constraints are too expensive [27] where practical algorithms require very efficient numerical solutions.

PROUD [30] and other algorithms [18], [24] minimize[3]

$$\min_{\mathbf{x}} \left\{ \sum_{i>j} a_{ij}(x_i - x_j)^2 : \mathbf{Hx} = \mathbf{b} \right\} \qquad (2)$$

where the *linear objective* is changed to the *quadratic objective*. The ease of minimization is due to the simplicity of the *quadratic objective* which results in the problem being solved via the solution of one system of linear equations.

Another algorithm, GORDIAN-L [28] and [9][4] minimizes the linear objective through a series of quadratic objectives with updated edge weights

$$\min_{\mathbf{x}}^{\nu} \left\{ \sum_{i>j} \frac{a_{ij}}{\left|x_i^{\nu-1} - x_j^{\nu-1}\right|} \left(x_i^{\nu} - x_j^{\nu}\right)^2 : \mathbf{Hx}^{\nu} = \mathbf{b} \right\} \qquad (3)$$

where $\mathbf{x}^{\nu-1}$ and $\mathbf{x}^{\nu}$ denote cell-location vectors at iterations $\nu - 1$ and ν. A quadratic objective is used to avoid the nondifferentiability of (1). Coefficients $\gamma_{ij} = a_{ij}/|x_i^{\nu-1} - x_j^{\nu-1}|$ are computed at a given iteration. The updated values x_i^{ν} are then found by quadratic minimization and used to recompute γ_{ij}.

As an alternative, regularization of (2) has been proposed in [1] and considers

$$\min_{\mathbf{x}} \left\{ \sum_{i>j} a_{ij}\sqrt{(x_i - x_j)^2 + \beta} : \mathbf{Hx} = \mathbf{b} \right\}. \qquad (4)$$

This optimization problem was solved in [1] in two ways, with a linearly-convergent fixed-point method due to Eckardt's [7], [8] generalization of the Weiszfeld algorithm [34], and with a novel primal-dual Newton method having quadratic convergence. Numerical testing in [1] illustrates the tradeoffs in values of $\beta > 0$ versus time and difficulty.[5]

It appears that PROUD favors a simpler but "incorrect" objective to capitalize on known algorithms, while GORDIAN-L pursues the "correct" objective with a new specialized algorithm. The regularization approach is in the middle—the objective function is modified by very little so that standard optimization techniques can be applied, although the optimization techniques are perhaps more involved than those in PROUD.

B. Delay Approximation

Performance-driven analytical placers typically represent critical path delays with an approximation of the Elmore delay model [10], but differ in how they address performance objectives into their formulations.

[2] In [2], the l_p norm $(\sum |x_i|^p)^{1/p}$ is regularized with $(\sum |x_i|^p + \beta)^{1/p}$, which is not smooth for $p = 1$ (the Manhattan norm which governs cell-to-cell distances used in wirelength and delay estimation).

[3] To further illustrate the purpose of the linear constraints, we point out that PROUD [30] do not include any constraints into the problem formulation; i.e., $\mathbf{H} = \mathbf{0}$. On the other hand, both [18], [24] include first moment constraints $(1/n)\sum_i A_i x_i = X_c$ to help cell spreading, where A_i is the area of cell i and X_c is the desired center of gravity.

[4] Another example of constraint selection: GORDIAN-L [28] includes the first moment constraints, similar to [18] whereas [9] abandons constraints in favor of an alternative approach to distribute cell areas based on Poisson's equation.

[5] It is also shown in [1] that the GORDIAN-L heuristic can be interpreted as a special case $\beta = 0$ of a fixed-point method having guaranteed linear convergence for $\beta > 0$. The proof consists of differentiating the objective function in (4), setting $\beta = 0$ and comparing to (3).

Timing-driven placement typically relies on a particular delay model for individual nets, and even "pin-to-pin" segments of interconnect. Based on an equivalent-Π model (i.e., a lumped-distributed model with half the capacitance at each end), the Elmore delay can be represented in a convenient form with posynomials, which allows a transformation into an optimization instance with strictly convex objective (see ([12], Sec. 6.8.2.3)). However, this technique is rather limited, e.g., it is not compatible with absolute values appearing in length calculations such as (3).

A closed-form expression for interconnect delay of a distributed *RC* line has been derived in [26] (see also [3], [4]). [14] simplified that model by decomposing the distance into x and y portions and ignoring the "crossterms" between x and y. Thus, the delay between cells i and j is represented as

$$\begin{aligned} d_{ij} = bd_i + \beta r_i C_i^L + \alpha(r_h c_h |x_i - x_j|^2 + r_v c_v |y_i - y_j|^2) \\ + \beta C_j (r_h |x_i - x_j| + r_v |y_i - y_j|) \\ + \beta R_i (c_h |x_i - x_j| + c_v |y_i - y_j|) \end{aligned} \tag{5}$$

where

- bd_i intrinsic cell delay of cell i;
- C_i input load capacitance;
- R_i equivalent on-resistance of the output transistor;
- C_i^L capacitive load.

r_h, r_v, c_h and c_v are per-unit vertical and horizontal resistance and capacitance.

[14] determined optimal values of constants experimentally as

- $\alpha = 1.02, \beta = 2.21$ for 90% rise threshold
- $\alpha = 0.59, \beta = 1.21$ for 70% rise threshold
- $\alpha = 0.5, \beta = 1.0$ for 62% rise threshold.

A popular *ad hoc* approach is to convert timing-analysis results into net weights [31], [25] used in the wirelength objective function. For example, SPEED [25] extends the GORDIAN-L algorithm to account for timing information by modifying the objective function in (3) as

$$\min_{\mathbf{x}^\nu} \left\{ \sum_{i>j} \omega_{ij} \gamma_{ij}^\nu \left(x_i^\nu - x_j^\nu\right)^2 : \mathbf{H}\mathbf{x}^\nu = \mathbf{b} \right\}$$

where ω_{ij} are net weights calculated during a timing-analysis step performed between iterations ν and $\nu - 1$. Known cell coordinates and a wiring model for each net are required for the timing analysis. For a net model, a star model is used with Elmore delays. Note that the weights do not directly account for the delay. Intuitively, the effect of weights is to duplicate important nets to increase their influence on the total [weighted] wirelength objective. Integer weight k is equivalent to k copies of the same net. We are not aware of convergence rate studies or nontrivial *a priori* conditions for convergence. In particular, convergence is unclear when intermediate optimizations are not solved to the true minimum, which is the case in real-life solvers.

Including path delays explicitly as nondifferentiable constraints is a more rigorous approach [13], [14], [29], but also more challenging for implementations and numerics. A typical implementation tradeoff is between, on one side, budgeting of net delays and including constraints for (many) individual nets into the optimization problem, or on the other side, establishing (fewer) constraints for individual critical paths. Per-net constraints are simple and only include the locations of incident cells, while per-path constraints include many more cells and lead to more elaborate and potentially less successful line search.

For example, the *Prime* algorithm [14] handles constraints using Lagrangian relaxation (one constraint per-path). The problem reduces to the minimization of

$$L(\mathbf{x}, \lambda) = \sum_{ij} c_{ij} |x_i - x_j| + \sum_{k=1}^{\pi} \lambda_k (h_k(x) - \tau)$$

where $h_k(\mathbf{x})$ is the path-delay function and τ is the target clock cycle. Rewriting gives the primal minimization objective[6]

$$l(\mathbf{x}) = \sum_{ij} \left[c_i j |x_i - x_j| + \sum_{k \in K_{ij}} \lambda_k d_{ij} \right] + \text{const.}$$

The resulting method is analogous with a nonlinear resistive-network problem. They also draw an equivalence relationship to GORDIAN-L [28]. The nonlinear problem is solved by using piecewise-linear approximations and explicitly monitoring for and skipping nondifferentiabilities. The duals are updated using a simple Newton method.

Following the other alternative, the *RITUAL* algorithm [29] computes required arrival r_i and actual arrival times a_j by performing timing analysis between iterations. Then it minimizes

$$L = 0.5\mathbf{w}^T \mathbf{Q} \mathbf{w} + \mathbf{b}^\mathbf{T} \mathbf{w}$$

(same as in *GORDIAN* [28] via $\mathbf{w} = (\mathbf{x}, \mathbf{y})$), subject to constraints

$$a_j \geq a_i + d_{ij} \quad a_j \leq T_e \quad a_i \geq T_s$$

where

$$d_{ij} = R_i * \sum_{\text{sinks}} (c_h |x_i - x_{\text{sink}}| + c_v |y_i - y_{\text{sink}}|).$$

The use of variables x_i and x_{sink} is reminiscent of linear programming (LP) formulations for bounding box minimization. This results in more constraints (four active constraints per net), compared to *Prime*. In order to avoid the numerical degeneracies of linear programming, *RITUAL* resorts to quadratic programming. Lagrangian relaxation is used to handle constraints; additionally it is shown that only "active constraints" (determined by graph traversal) need to be considered. The primal problem reduces to solving one linear system, and the duals are solved by use subgradient optimization.

[27] reduces a series of delay budgeting problems to linear programming by a general construction that approximates nonlinear delay terms by univariate piecewise-linear functions. The resulting LP formulations are too costly to solve since their

[6]The constant term is the sum of τ, intrinsic cell delay and delay from the solution in the y-direction.

combinatorial complexity increases when better precision is required.[7]

These approaches to including explicit performance information, all have considerable drawbacks and implementation difficulties, e.g., the degeneracies of linear programming formulations are rather difficult to address in a simple implementation. Our approach will be to approximate linear-programming formulations by nonlinear programming to allow efficient Newton methods. The computational complexity of our proposed smooth nonlinear approximations does not increase with required precision, unlike the complexity of piecewise-linear approximations that enable linear programming (cf. [27]). Therefore, when solving mixed linear and nonlinear programs, we pursue smooth nonlinear approximations rather than reductions to linear programming.

We distinguish two desired features of such an approach. These are : 1) strictly convex edge delay estimates in terms of cell positions; and 2) their regularization to remove nondifferentiabilities. Critical-path delays are then computed as sums of corresponding edge delays, resulting in smooth and convex functions.

We demonstrate below how these elements lead to concise performance-driven formulations amenable to efficient Newton-type methods, where path delays are integrated into the objective function.

III. Function Regularization

In this section, we develop a general method to modify functions so that standard-optimization techniques can be applied. The real-valued function $f(\cdot)$ that we modify is assumed to be convex and continuous over an open subset $X \subset \mathbf{R^n}$. We seek a family of smooth convex functions $f_\beta(\cdot)$ for $\beta > 0$ with

1) $\lim_{\beta\to 0} f_\beta(\mathbf{x}) = f(\mathbf{x})$ uniformly on $\mathbf{R^n}$;
2) $\lim_{\beta\to 0} \inf_{\mathbf{x}\in\mathbf{R^n}} f_\beta(\mathbf{x}) = \inf_{\mathbf{x}\in\mathbf{R^n}} f(\mathbf{x})$.

For simple functions, we provide "recipes" for regularization and prove their desired limit behavior. For more complicated functions, e.g., $f(x) = 2|x| + x^2$, we isolate nondifferentiabilities to small symbolic fragments (e.g., absolute value functions) for which recipes exist. Replacing the symbolic fragments with their regularizations yields a regularization of the overall function.

A. Piecewise-Linear Functions

We begin by considering $f : \mathbf{R} \to \mathbf{R}$ and distinguish a common case where regularizing f is easy

$$f(x) = \begin{cases} \chi_1(x - x_0) + C, & \text{if } x \geq x_0 \\ \chi_2(x - x_0) + C, & if x < x_0 \end{cases} \tag{6}$$

where $\chi_1 > 0, \chi_2 < 0$, and C is arbitrary, i.e., a "V-shape".

For $p \geq 2$, the β-regularization of f is defined by

$$\forall x, f_\beta(x) = C + (|f(x) - C|^p + \beta)^{\frac{1}{p}}. \tag{7}$$

[7]Reference [27] relies on specifics of particular budget delay formulations to apply rather elaborate algorithms based on min-cost flows and graph-based simplex methods.

This regularization is different from that in [20], as p is now a regularization parameter.

Example 1: If $f(x) = |x|$ and $p = 2$ then $f_\beta(x) = \sqrt{x^2 + \beta}$. This can be used to regularize the l_1-norm. The value $p = 2$ is typical since it is the smallest value for which the regularized function is twice-differentiable.

Theorem 1: $\forall p \geq 2, \forall \beta > 0, f_\beta$ defined in (7) is:

1) at least $(\lceil p \rceil - 1)$-times continuously differentiable;
2) strictly convex;
3) for $p > 2$ a noninteger: exactly $\lceil p \rceil - 1 = \lfloor p \rfloor$-times continuously differentiable;
4) for $p \geq 2$ an integer: at least p-times differentiable (in fact, infinitely differentiable) *iff* $\chi_1 = -\chi_2$.

Proof: The function f_β is continuous everywhere and is infinitely differentiable everywhere except at x_0. The second derivative exists and is positive everywhere, except possibly at x_0, leading to **2)**.

The 1st, $\ldots, (\lceil p \rceil - 1)$th left and right derivatives of $|f(x) - C|^p$ are all zero at x_0, while for p a noninteger, its higher derivatives are infinite. The chain rule then implies **1)** and **3)**.

If $\chi_1 = -\chi_2$ and $p \geq 2$ is an integer, then $f_\beta(x) = C + ((\chi_1)^p|x - x_0|^p + \beta)^{1/p}$, and is infinitely differentiable. If, instead, $\chi_1 \neq -\chi_2$, then the left and right pth derivatives of $|f(x) - C|^p$ differ at x_0, proving the "only if" of **4)**. □

Theorem 2: For f_β defined in (7), we have:

1) $\forall x, |f_\beta(x) - f(x)| \leq \beta^{1/p}$;
2) $\lim_{\beta\to 0} f_\beta(x) = \lim_{p\to\infty} f_\beta(x) = f(x)$ uniformly on $\mathbf{R}$;
3) $\forall x, \forall \beta_1 > \beta_2 > 0, f_{\beta_1}(x) > f_{\beta_2}(x) > f_0(x) = f(x)$;
4) $\lim_{\beta\to 0} \min_{x\in\mathbf{R}} f_\beta(x) = \min_{x\in\mathbf{R}} f(x)$;
5) $\forall \beta > 0 \lim_{x\to\pm\infty} f_\beta(x) = f(x)$.

Proof: The inequalities **3)** are shown by subtracting C and taking both sides of the inequality to the power p. Note that $f_\beta - C \geq \beta^{1/p} > 0$ for $\beta > 0$.

For **1)**, observe that $\beta = |(f_\beta - C)^p - (f - C)^p| = |(f_\beta - C) - (f - C)| \cdot |(f_\beta - C)^{p-1} + \cdots + (f - C)^{p-1}| \geq (f_\beta - C)^{p-1}|f_\beta - f| \geq \beta^{p-1/p}|f_\beta - f|$.

Item **2)** follows from **1)**. **4)** follows from **2)** and **3)**.

Using **3)** and the above inequalities, we have $|f_\beta(x) - f(x)| \leq \beta/(f_\beta(x) - C)^{p-1}$. Since $\lim_{x\to\pm\infty} f_\beta(x) = \infty$ for $\chi_1, \chi_2 \neq 0$, we have that $\lim_{x\to\pm\infty} |f_\beta(x) - f(x)| = 0$, proving **5)**.

Note that Theorem 2, part **5)** is not true anymore with $\chi_1 = 0$ or $\chi_2 = 0$ (e.g., $f(x) = \chi_1 \max\{x, 0\}$ or $f(x) = -\chi_2 \min\{x, 0\}$). These two cases can be reduced to $\chi_1 \neq 0$ and $\chi_2 \neq 0$ by rotating the plot around the coordinate center, which motivates an alternative regularization of f that is coordinate-independent. Consider the upper branch of a hyperbola with asymptotes going along the plot of $f(x)$. For $f(x) = \chi_1 \max\{0, x - x_0\} + C$ (the same as (6) with $\chi_2 = 0$) such a hyperbola can be defined in the x-y plane (i.e., for $y = f(x)$) with the following equation: [8]

$$(2(y - C)/\chi_1 - x + x_0)^2 - (x - x_0)^2 = \beta. \tag{8}$$

[8]For $\chi_1 > 0$ and $\chi_2 < 0$, a hyperbola similar to that in (8) would define an infinitely differentiable regularization, otherwise satisfying the statements of Theorems 1 and 2. For $\chi_1 \neq -\chi_2$, it will differ from the regularization defined in (6) because the latter is not twice-differentiable according to Theorem 1 part **4)**.

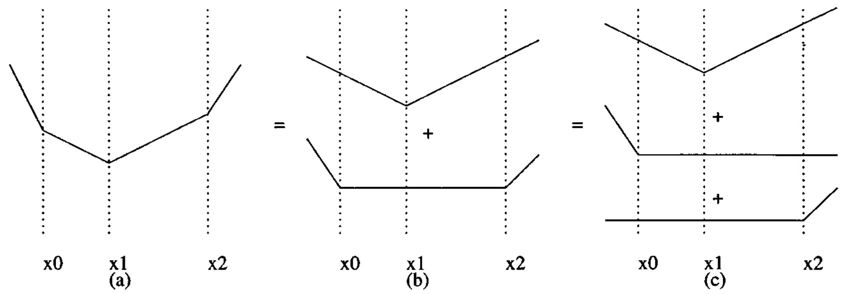

Fig. 1. Illustration of the proof of Theorem 3. (a) The original piecewise-linear function with three break points., (b) The same function divided into the sum of a simple function and a piecewise-linear function with two break points. (c) The final result in which the original function in (a) is divided into three simple functions obtained by the division of the piecewise-linear function with two break points in (b).

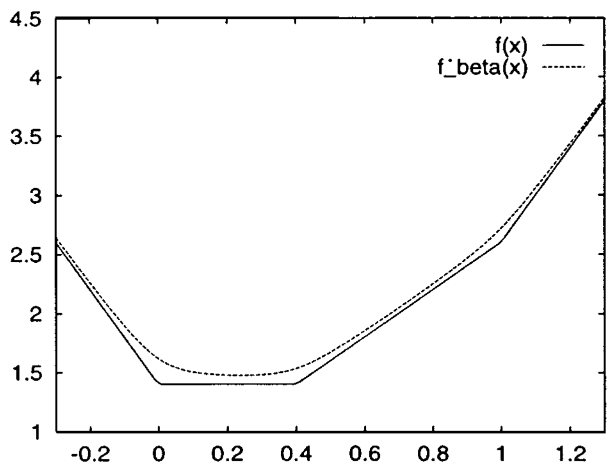

Fig. 2. $f(x) = |1-x| + 2|x| + |0.4-x|$ with $p = 2$, $\beta = 0.01$ and $f_\beta(x) = \sqrt{(1-x)^2+\beta} + 2\sqrt{x^2+\beta} + \sqrt{(0.4-x)^2+\beta}$.

One can verify that Theorem 1 holds for this regularization with the regularized function being infinitely differentiable as well. Theorem 2 also holds.

The asymptotic behavior of regularizations stated in Theorems 1 and 2 can be proven for the case of arbitrary univariate convex piecewise-linear functions using the following theorem. □

Theorem 3: Any convex piecewise-linear function with k linear segments can be represented (not necessarily uniquely) as a sum of a constant and $k-1$ convex functions of the form (6), possibly with $\chi_1 = 0$ or $\chi_2 = 0$

Proof: We call a convex function of the form given in (6) a *simple function*. It suffices to show that a convex piecewise-linear function $f(x)$ with $k-1 > 0$ break points can be represented as a sum of such a function with $k-2$ break points and a simple function.

Take a break point x_0 and consider the two adjacent linear segments. Continuing them to $+\infty$ and $-\infty$ gives a simple function $g(x)$ such that $g(x) \le f(x)$. Moreover, $f(x) - g(x)$ is convex since we are simply subtracting a linear function from $f(x)$ on each of $(-\infty, x_0]$ and $[x_0, +\infty)$, while $f(x) - g(x)$ has a horizontal linear segment in place of the two maximal linear segments adjacent to x_0 and is convex near x_0. The procedure is illustrated in Fig. 1. □

Corollary 4: Any univariate convex piecewise-linear function with k linear segments can be β-regularized with $|f(x) - f_\beta(x)| \le \beta^{1/p}(k-1)$. The regularization will possess properties according to Theorems 1 and 2.

Example 2: If χ_1 and χ_2 in (6) are of the same sign, but $f(x)$ is still convex, then by Corollary 4 it can be regularized.

Fig. 2 shows a plot of piecewise-linear $f(x)$ together with $f_\beta(x)$. $f(x)$ can be interpreted as the objective function for a placement problem with one movable cell connected to three fixed pads located at 1, 0 and 0.4 by edges of weight 1,2, and 1, respectively.

A single univariate piecewise-linear function can be efficiently minimized with convex binary search, but this becomes difficult when such functions are combined, e.g., as $f(g(y)) + f(x)g(y)$, since the combinatorial complexity (i.e., the number of maximal domains of linearity) considerably increases. Working with smooth approximations, combinatorial problems can be circumvented via symbolic differentiation. In other words, gradient and Hessian computations and smooth minimization may be much easier than solving linear programs.

We now show a generalization of the above theorem to multivariate functions.

Lemma 5: An arbitrary multivariate convex piecewise-linear function with k linear domains can be represented as a maximum of k linear functions.

Proof: Every linear domain is represented by a linear function, which is dominated by the original function according to the convexity property. Since the domain of the original function is the union of linear domains, the maximum of all respective linear functions is never smaller than the original function. □

Corollary 6: An arbitrary multivariate convex piecewise-linear function segments can be β-regularized with $|f(x) - f_\beta(x)| \le \beta^{1/p} \log k$. The regularization will possess properties according to Theorems 1 and 2.

Proof: The maximum function of k arguments can be regularized by a reduction to k two-argument maximum functions, arranged into a binary tree of depth $\log k$. □

Note that the estimate in Corollary 6 is better than in the previously considered special case of univariate functions.

Corollary 6 shows that an arbitrary convex piecewise-linear function can be minimized using β-regularization. Additionally, we can regularize an arbitrary linear program. First, by a well-known reduction (used in the Ellipsoid method) it suffices to solve constraint-satisfaction for $A\mathbf{x} \le \mathbf{b}$. The latter can be reduced to unconstrained minimization of a convex piece-

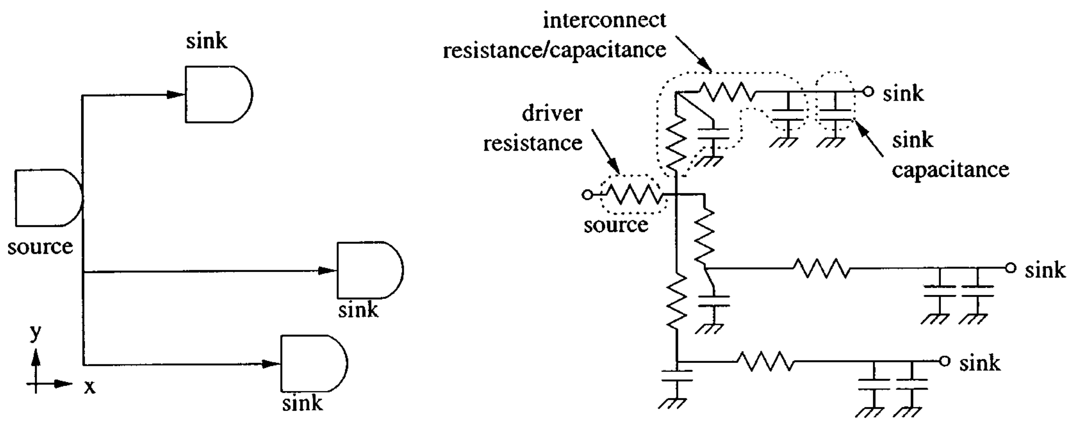

Fig. 3. Wiring model for a single net.

wise-linear function by minimizing the violation of constraints (the constraints are satisfied if the value 0 is reached). However, the practical utility of such a general reduction is not clear.

B. Symbolic Regularization and Examples

For many functions, the cusps that need to be regularized are due to an absolute value or more general case analysis in the symbolic representation of the function. Define F by

$$\forall x, F(x) = \begin{cases} F_1(x-x_0)+C, & \text{if } x \geq x_0 \\ F_2(x-x_0)+C, & \text{if } x < x_0 \end{cases} \quad (9)$$

with $F_1(t)$ continuously differentiable for $t \geq 0$ and $F_2(t)$ continuously differentiable for $t \leq 0$. We also assume that $F(t)$ is nonnegative and convex as well as $F_1(0) = F_2(0) = 0$ (but possibly $F_1'(0^+) \neq F_2'(0^-)$). Let C be arbitrary.

For $p \geq 2$, the β-regularization of $F(x)$ is defined by

$$\forall x, F_\beta(x) = C + (|F(x)-C|^p + \beta)^{1/p} \quad (10)$$

which subsumes (7) for the piecewise-linear case.

Replacing a symbolic fragment with a regularization in a larger function leads to a smooth function. Multiple regularizations can be performed with single β or multiple independent β_i. The convexity properties and the limit behavior of the fragment regularizations often extend to the resulting function through sums, products, exponents, etc. Properties 1 and 2 from [20] provide excellent examples of such symbolic regularization; however, the nondifferentiable fragments are regularized differently there.

Example 3: $\max\{a,b\} = (a+b+|a-b|)/2$ and $\min\{a,b\} = (a+b-|a-b|)/2$ can be regularized as, respectively, $(a+b+(|a-b|^p+\beta)^{1/p})/2$ and $(a+b-(|a-b|^p+\beta)^{1/p})/2$.

In particular, for $f(x) = \max\{0,(x-x_0)\}$ and $p=2, f_\beta(x) = 1/2((x-x_0)+\sqrt{(x-x_0)^2+\beta})$, which matches the hyperbola in (8) when $\chi_2 = 1$.

Example 4: $f(a,b) = \max\{(a+b)^2,(a-b)^2\} = a^2+b^2+2|ab|$ can be regularized as $f_\beta(a,b) = a^2+b^2+2\sqrt{a^2b^2+\beta}$.

C. Practical Issues

When $f(x)$ is convex, but not strictly convex, it can have multiple minimizers. However, $f_\beta(x)$ is strictly convex for $\beta > 0$ and has only one minimizer (see, e.g., Fig. 2). From the theorems in Section III, under mild conditions a minimizer of f can be obtained as the limit of the minimizer of $f_\beta(x)$ as $\beta \to 0$. In some cases, the minimizer of $f_\beta(x)$ already minimizes $f(x)$, e.g., for any β the unconstrained minimizer of (7) is the unconstrained minimizer of (6) if $\chi_1 > 0$ and $\chi_2 < 0$.

Numerical methods using β-regularization require specific values of β to evaluate the regularization or its derivatives. Ideally, there should be a way to define β via an instance- and scale-independent parameter (clearly, if one scales all x coordinates in an instance up by a factor of 100, the effect of the old value of β will be different). In practice, the pth exponent of the maximal x value for a problem can be multiplied by an instance-independent β_0 to produce β.

IV. Applications to Analytical Placement

We now show how regularization enables the use of Newton-type methods in delay minimization. With a judicious choice of a wiring model for each net, the application of regularization results in smooth and convex path-based delay modeling which previous analytical placers were unable to handle *directly* in the optimization process (cf. Section 2.2).

A. Convex Delay Model for a Single Net

For a given net, we use rectilinear L-shaped interconnects to connect the source directly to each sink (see Fig. 3). An equivalent-L model is used for each such segment L-segment by distinguishing the x- and the y-leg. Let C_s be the capacitance of sink s, R_d be the driver resistance and r_x (r_y) and c_x (c_y) be the per-unit interconnect series resistance and shunt capacitance, respectively, in the x- (y-) direction equivalent-L model. The delay from the source to a specific sink consists of:

1) Source resistance times all downstream capacitance

$$R_d(\sum_s c_x|\chi| + c_y|\gamma| + C_s). \quad (11)$$

2) Interconnect resistance times sink capacitance:

$$(r_x|\chi| + r_y|\gamma|)C_s. \quad (12)$$

3) Interconnect resistance times interconnect capacitance

$$r_x c_x \chi^2 + r_y c_y \gamma^2 + r_y c_x |\chi||\gamma| \quad (13)$$

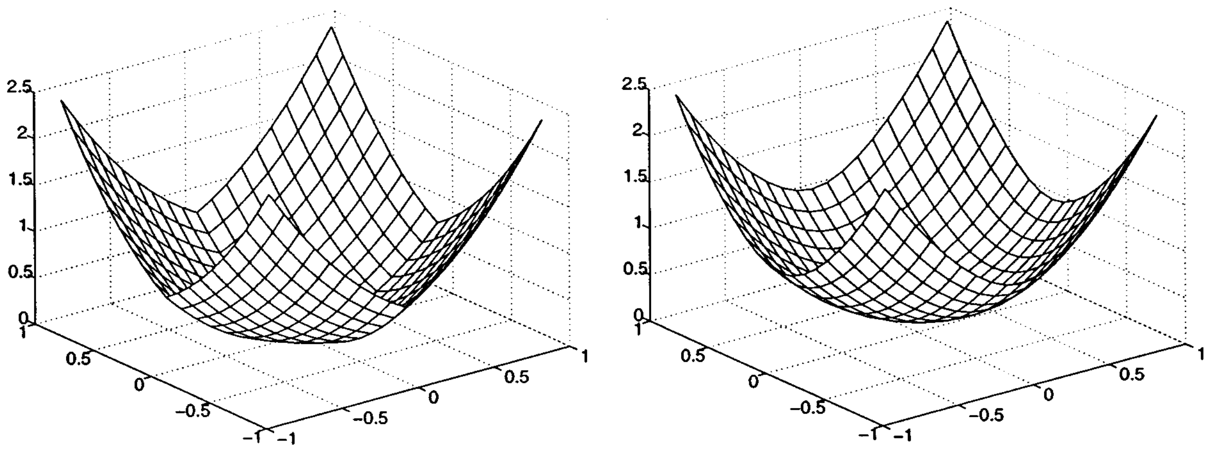

Fig. 4. Regularization of the interconnect delay.

where $\chi = x_d - x_s$ and $\gamma = y_d - y_s$. Components **1)** and **2)** are convex. However, they are nondifferentiable when cells are aligned vertically or horizontally. Component **3)** is clearly convex if (but not "only if") the cross term $|\chi||\gamma|$ is ignored. The magnitude of the cross term can be comparable to other terms and should not be ignored if strict convexity can be otherwise guaranteed. Note that the L-shaped interconnect is assumed to have its vertical leg incident to the source, and its horizontal leg incident to the sink.[9]

This model is somewhat similar to the model introduced in [14] [See (5)]. Terms representing the driver-resistance times all downstream capacitance and the interconnect-resistance times sink capacitance are the same except for the constant factors that [14] determined experimentally (those can also be trivially added to our model and do not influence theoretical analyses). On the other hand, we do not ignore the coupling of x and y directions in the interconnect delay term. [14] also notes that dropping the interconnect term results in a linear model and reduces to that used by Jackson and Kuh [13].

Proposition 7: When $c_y/r_y > 0.25c_x/r_x$ then the delay component in Formula (13) is strictly convex.

Proof: The functions $q_-, q_+ : \mathbf{R}^2 \to \mathbf{R}$

$$q_-(\chi,\gamma) = \chi^2 r_x c_x + \gamma^2 r_y c_y - r_y c_x \chi\gamma \quad (14)$$
$$q_+(\chi,\gamma) = \chi^2 r_x c_x + \gamma^2 r_y c_y + r_y c_x \chi\gamma \quad (15)$$

are strictly convex under the assumption of the proposition. This claim is easily checked by considering the nonnegativity of the Hessians of q_- and q_+, i.e., $(2r_xc_x)\cdot(2r_yc_y) > (r_yc_x)\cdot(r_yc_x)$ for both functions. Since $|r_yc_x| = \max\{r_yc_x, -r_yc_x\}$, the function $d(\chi,\gamma) = \max\{q_-(\chi,\gamma), q_+(\chi,\gamma)\}$ is the maximum of two strictly convex functions and is hence strictly convex. The function $d(x_d - x_s, y_d - y_s)$ is strictly convex as a composition and equals component (c) of the interconnect delay. □

Thus, given suitable capacitance to resistance ratios in the different routing directions (e.g., different metal layers), the cross term can be kept. Regularization of delay components **1)** and **2)** is straightforward. Component **3)** is regularized similarly to Example 4 in Section III.B as

$$\sqrt{(\chi\gamma r_y c_x)^2 + \beta} + \chi^2 r_x c_x + \gamma^2 r_y c_y. \quad (16)$$

and the convexity is preserved. Fig. 4 illustrates (13) and its regularization (16).

Again, observe that proposed wiring model assumes an order in which the x- and y-legs of the L-segment are routed (when the source and the sink do not have the same x or y coordinates). However, we select this order arbitrarily and adjust the resistance-capacitance ratios in Proposition 7 appropriately. Certainly, both models could be represented for each L-segment during the optimization, e.g., the average or the maximum of the two models can be easily expressed (with maximum function being regularized). Quite likely, a single wiring model represents typical tradeoffs accurately enough, e.g., in comparison with net re-weighting or various heuristic updates used by earlier proposed methods. □

B. Wirelength and Delay Brought Together

We have illustrated the use of function regularization to both wirelength and delay approximation. Here, we propose several concise performance-driven formulations which tie wirelength and delay approximations together in a consistent fashion; i.e., a problem formulation in which both wirelength and delay are considered simultaneously. In order to be specific, these formulations use explicit enumerations of critical paths, potentially unacceptable for large modern circuits (there may be exponentially many critical paths). We note, however, that β-regularization advocated in this work can be applied with equal success to timing-driven optimizations that handle critical paths implicitely, via Static timing analysis. This is because the delay of each timing edge can be regularized independently, and their sum will be a regularization of path delay.

Next, we propose unconstrained formulations which are favored by newer analytical placers [9] and can be transparently handled by leading nonlinear minimization algorithms and software [12], [16], [35].

[9]This seemingly arbitrary choice can be "fixed" in several ways, but we feel that the small overall improvement in accuracy will not justify increased complexity.

TABLE I
QUADRATIC PLACEMENT AND β-REGULARIZATION COMPARED AGAINST OPTIMAL LINEAR PROGRAMMING RESULTS. TOTAL WIRELENGTH (x- AND y-) AND CPU SECONDS AVERAGED OVER THE x- AND y-RUNS ARE REPORTED FOR ALL METHODS. CPLEX 6.5.1 RUNS WERE PERFORMED ON AN IBM RS/6000 3CT WORKSTATION THAT MEASURED APPROXIMATELY 1.5 TIMES SLOWER THAT THE SUN ULTRA-10/300 MHz WORKSTATION WHERE ALL OTHER RUNS WERE PERFORMED

Sub-circuit description				Solution methods					
Instance name	Nodes fixed	Nodes movable	Edges	CPLEX 6.5.1 WL	CPLEX 6.5.1 CPU⊙	β-regularization WL	β-regularization CPU⊙	Quadratic WL	Quadratic CPU⊙
test1	2191	3295	10636	2.42e6	28.14sec	2.61e6	11sec	3.05e6	2.3sec
test2	2155	6739	17568	3.39e6	4min	3.87e6	19sec	2.16e6	12 sec
test3	6545	17380	53489	1.68e7	15.68min	1.95e7	61sec	2.24e7	36 sec

For *penalization of path delays in a particular set P* (e.g., specified by the designer) the objective function is

$$\min_{\mathbf{x},\mathbf{y}} \left\{ f(\mathbf{x},\mathbf{y}) + K \sum_{\pi\in P} d_\beta^\pi(\mathbf{x},\mathbf{y}) \right\} \tag{17}$$

where $f(\mathbf{x},\mathbf{y})$ is an estimate of the total netlist wirelength (e.g., (4)) and $d_\beta^\pi(\mathbf{x},\mathbf{y})$ is the regularized delay for path π. Parameter K normalizes delay and wirelength, thus allowing one to tune the trade-off between smaller wirelength and smaller delay. P may be constructed to exclude false paths and include paths that look important to circuit designers.

For *minimization of longest path delays* the objective function is

$$\min_{\mathbf{x},\mathbf{y}} \left\{ f(\mathbf{x},\mathbf{y}) + K \textstyle\sum_{\pi\in P} \max\left\{ \zeta, d_\beta^\pi(\mathbf{x},\mathbf{y}) \right\} \right\} \tag{18}$$

where ζ is a "soft" target delay, i.e., we are not interested in minimizing path delays below ζ. Since delay information is included in the objective function, the number of constraints is *not increased* by the inclusion of performance information.

The parameter ζ can be adjusted between iterations of timing optimization. Note that this formulation avoids computing *the longest* path, as the multi-variate max function is not smooth and has exponentially many cusps. Alternatively, the multivariate max can be rewritten as a chain of two-variate max functions and regularized sequentially ([17] proposes a new, direct regularization of the multivariate max function). Also, note that when manual circuit modifications are allowed after automatic CAD tools, it is usually not difficult to speed up any given path. Therefore, minimizing the longest path delay is not the only goal—one may want to minimize the number of critical paths, or the average criticality of a path. The latter can be achieved by our formulation with appropriate selection of the coefficient K, since average path delay is a special case of weighted total path delay.

Finally, although we favor unconstrained formulations, we note that convex constraints, e.g., first moment constraints [18], [28], [25], [24] can be easily included and solved [35] with Newton-type methods.

V. EMPIRICAL VALIDATION

In this section, we compare the placement quality produced by function smoothing techniques (we used β-regularization with $\beta_0 = 0.01$), by straightforward quadratic placement[10] and by linear programming (guaranteed optimality, but with huge running time).[11] We are not aware of such a comparison in the literature, even though quadratic placement has been widely used in academic and industrial placement tools.

We considered three subcircuits that appear during top-down placement of larger circuits and converted them into graphs using a standard clique conversion so that all approaches be equally applicable. During top-down placement [2], [32], a given circuit is recursively partitioned in two by straight-line cuts until the partitions are so small, that all cells inside can be placed by exhaustive enumeration or branch-and-bound. Clearly, the core of the algorithm is the partitioning step. If one subcircuit is partitioned at a time, all other cells are considered in the geometric centers of their respective subregions. Placement is performed to minimize wirelength, and its results can be interpreted, e.g., by median partitioning (see [32] for a more complicated scheme). The main difficulty that can be addressed by our analytical formulation is the ability to trade of wirelength with other, possibly nonlinear, objectives. Our experiments test the practicality of minimizing the wirelength alone, since: 1) the timing terms in our formulations are often not nearly as computationally intensive as wirelength; and 2) the structure of timing-critical paths may vary considerably, so that even a dozen specific experiments are unlikely to give new insight compared to the pure minimization of wirelength.

It is important to note that, unlike in min-cut partitioning, analytical placement requires sufficiently many fixed vertices to "pull apart" the movable vertices. The top-down placement paradigm does not provide such instances at the first level of recursive partitioning. For example, our largest testcase with 17 380 movable cells is a part of a larger circuit having 68 685 placeable cells (roughly four times more) and 744 fixed pads. This corresponds to the third level of top-down partitioning. The first three levels were performed by a min-cut partitioner.

Note that, since a very large number of partitionings (min-cut or analytical) need to be solved in order to place a circuit, each needs to be performed extremely fast. For example, in pure min-cut placemet (which cannot directly address nonlinear terms), competitive runtimes for a circuit with 20 K cells will be on the order of 10–20 s per start (2–4 independent randomized starts are often used in placement applications). Analytical placers are typically slower, but their runtime must have the same order of magnitude for them to stay practical. Note that

[10] That is, solving one linear system per x- or y-dimension.

[11] To avoid accounting of the effects of converting multi-pin nets into cliques or stars, we apply all methods after such a conversion. Optimal solutions produced by linear programming give a basis for comparison.

analytical placers are typically deterministic and are therefore applied only once to each problem instance.

Our empirical results are reported in Table I. Clearly, the results may vary for other applications, and also depend on specific values of β and convergence criteria used. In fact, the latter would normally be tuned to particular applications and runtime budgets (the smaller the β, the better solutions can be produced, for the cost of larger runtime). Therefore, our experiments should only be considered as a "proof of concept".

For the benchmarks considered, our implementation of β-regularization is able to quickly find solutions that are within 20% of optimum produced by the leading commercial linear programming solver CPLEX 6.5.1 (March'99 revision). The CPU time growth for the β-regularization approach is very close to linear in the number of movable cells. Additionally, linear programming becomes inapplicable once the placement objective has nonlinear terms. This is often the case in leading-edge applications, because piecewise linearization of the nonlinear terms will significantly exacerbate run time.

Solutions produced by quadratic placement are at least twice as far from the optimum. Our results confirm that quadratic placement suffers from minimizing a wrong objective, no matter how well and how quickly this objective can be minimized in its own right.

In addition to the particular application we used to evaluate our methods, recent placement literature offers a good selection of applications and approaches of different kinds compatible with our methods. In particular, while [32] presents a top-down placer based on analytical optimization and avoids any use of min-cut partitioning, two works published in the Fall of 2000, develop approaches that do not use any form of partitioning: [22] proposed a flat force-directed macro cell placer, and [5] described a multilevel large-scale placer based on recent advances in numerical analysis. These two works suggest another type of application where our techniques may be useful.

VI. Conclusion

The main assertion of our work is that the mathematical properties of the objective function are more fundamental than particular optimization algorithms. The construction or adaptation of optimization algorithms may be simplified once the properties—especially differentiability and convexity—are understood.

Numerical solvers perform best with smooth and convex functions. However, nondifferentiable points occur in important objective functions, e.g., when Manhattan distances are used. We have presented general, provably-good regularization techniques for eliminating cusps in optimization objectives and discussed their theoretical and practical properties. In particular, we can apply nonlinear optimization to minimize an arbitrary convex piecewise-linear function (a widely-used abstraction for modeling electrical networks [19]), and even an arbitrary linear program. Our techniques rely on generic minimization algorithms rather than highly specialized heuristics, e.g., in previous works on circuit placement [25], [28]–[30]. The utility of our methods has been demonstrated for wirelength- and delay-based objectives in VLSI applications, where they lead to smaller and easier performance-driven placement formulations. Additionally, since convexity plays an important role in regularizations and subsequent optimization, we highlight the need in provably convex delay models. Our proposed Elmore-type delay model is similar to that used in [14] and is provably convex given conditions found in typical applications.

Acknowledgment

The authors wish to thank Dr. D. Morton of the University of Texas, Austin, for computational resources used in the preparation of the results of Section VI.

References

[1] C. J. Alpert, T. F. Chan, D. J. H. Huang, A. B. Kahng, I. L. Markov, P. Mulet, and K. Yan, "Faster minimization of linear wire length for global placement," *Proc. ACM/IEEE Int. Symp. Physical Design*, pp. 4–11, 1997.

[2] A. E. Caldwell, A. B. Kahng, and I. Markov, "Can recursive bisection alone produce routable placements?," in *Proc. Design Automation Conf. 2000*, pp. 477–482.

[3] D. L. Carter and D. F Guise, "Analysis of signal propagation delays and chip level performance due to on chip interconnections," in *Proc. Int. Conf. Computer Design*, 1983, pp. 218–221.

[4] E. H. Horneber and W. Mathis, "A closed-form expression for signal delay in CMOS-driven branched transmission lines," in *Proc. VLSI'87*, C. H. Sequin, Ed., pp. 353–362.

[5] T. F. Chan, J. Cong, T. Kong, and J. R. Shinnerl, "Multilevel optimization for large-scale placement," in *Proc. Int. Conf. Computer-Aided Design 2000*, pp. 171–176.

[6] C. Charalambous, "Acceleration of the HAP approach for the multifacility location problem," *Nav. Logist. Q.*, vol. 32, pp. 373–389, 1985.

[7] U. Eckhardt, "On an optimization problem related to minimal surfaces with obstacles," in *Optimization and Optimal Control, Lecture Notes in Mathematics 477*, R. Bulirsch, W. Oetti, and J. Stoer, Eds. New York: Springer-Verlag, 1975, pp. 95–101.

[8] ——, "Weber's problem and Weiszfeld's algorithm in general spaces," *Math.Prog.*, vol. 18, pp. 186–196, 1980.

[9] H. Eisenmann and F. M. Johannes, "Generic global placement and floorplanning," *Proc. ACM/IEEE Design Automation Conf.*, pp. 269–274, 1998.

[10] W. C. Elmore, "The transient response of damped linear networks with particular regard to wideband amplifiers," *J. Appl. Phys.*, vol. 19, no. 1, pp. 55–63, 1948.

[11] J. W. Eyster, J. A. White, and W. W. Wierwille, "On solving multifacility location problems using a hyperboloid approximation procedure," *Trans. Amer. Inst. Ind. Eng.*, vol. 5, pp. 1–6, 1973.

[12] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, U.K.: Academic, 1981.

[13] M. A. B. Jackson and E. S. Kuh, "Performance-driven placement of cell based IC's," *Proc. ACM/IEEE Design Automation Conf.*, pp. 370–375, 1989.

[14] T. Hamada, C. K. Cheng, and P. M. Chau, "Prime: A timing-driven placement tool using a piecewise-linear resistive network approach," *Proc. ACM/IEEE Design Automation Conf.*, pp. 531–536, 1991.

[15] M. Held, P. Wolfe, and H. P. Crowder, "Validation of subgradient optimization," *Math. Prog.*, vol. 6, pp. 62–88, 1974.

[16] C. Kelley, *Iterative Methods for Linear and Nonlinear Equations, Frontiers in Applied Mathematics*. Philadelphia, PA: SIAM, 1995, vol. 16.

[17] A. A. Kennings and I. L. Markov, "Analytical minimization of half-perimeter wirelength," presented at the *Proc. Asia and South Pacific Design Automation Conf. 2000*.

[18] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich, "GORDIAN: VLSI placement by quadratic programming and slicing optimization," *IEEE Trans. Computer-Aided Design*, vol. CAD-10, pp. 356–365, Mar. 1991.

[19] D. M. W. Leenaerts and W. M. G. van Bokhoven, *Piecewise-Linear Modeling and Analysis*. Boston, MA: Kluwer Academic, 1998.

[20] R. F. Love and J. G. Morris, "Solving constrained multifacility location problems involving l_p distances using convex programming," *Oper. Res.*, vol. 23, pp. 581–587, 1975.

[21] I. I. Mahmoud, K. Asakura, T. Nishibu, and T. Ohtsuki, "Experimental appraisal of linear and quadratic objective functions effect on force directed method for analog placement," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. 4(E77-A), pp. 710–725, 1994.
[22] F. Mo, A. Tabbara, and R. K. Brayton, "A force-directed macro-cell placer," in *Proc. Int. Conf. on Computer-Aided Design 2000*, pp. 177–180.
[23] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. New York: Academic, 1970.
[24] P. N. Parakh, R. B. Brown, and K. A. Sakallah, "Congestion driven quadratic placement," *Proc. ACM/IEEE Design Automation Conf.*, pp. 275–278, 1998.
[25] B. M. Riess and G. G. Ettelt, "Speed: Fast and efficient timing driven placement," *Proc. IEEE Int. Symp. Circuits and Systems*, pp. 377–380, 1995.
[26] T. Sakurai, "Approximation of wiring delay in MOSFET LSI," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 418–426, Apr. 1983.
[27] M. Sarrafzadeh, D. Knoll, and G. Tellez, "Unification of budgeting and placement," in *Proc. DAC'97*, pp. 758–761.
[28] G. Sigl, K. Doll, and F. M. Johannes, "Analytical placement: A linear or quadratic objective function?," *Proc. ACM/IEEE Design Automation Conf.*, pp. 57–62, 1991.
[29] A. Srinivasan, K. Chaudhary, and E. S. Kuh, "RITUAL: A performance driven placement for small-cell ICs," in *Proc. Int. Conf. Computer-Aided Design*, 1991, pp. 48–51.
[30] R. S. Tsay and E. Kuh, "A unified approach to partitioning and placement," *IEEE Trans. Circuits Syst.* , vol. 38, pp. 521–633, May 1991.
[31] R. S. Tsay and J. Koehl, "An analytical net weighting approach for performance optimization in circuit placement," *Proc. ACM/IEEE Design Automation Conf.'91*, pp. 620–625.
[32] J. Vygen, "Algorithms for large-scale flat placement," *Proc. ACM/IEEE Design Automation Conf.*, pp. 746–751, June 1997.
[33] B. X. Weis and D. A. Mlynski, "A new relative placement procedure based on MSST and linear programming," in *Proc. ISCAS'87*, pp. 564–567.
[34] E. Weiszfeld, "Sur le point pour lequel la somme des distances de n points données est minimum," *Tôhoku Math. J.*, vol. 43, pp. 355–386, 1937.
[35] S. J. Wright, *Primal-Dual Interior-Point Methods*. Philadelphia, PA: SIAM, 1996.

Ross Baldick (S'90–M'91) received the B.S. degree from the University of Sydney, Sydney, Australia, and the M.S. and Ph.D. degrees from the University of California, Berkeley.

He is an Associate Professor of electrical and computer engineering at the University of Texas, Austin.

Andrew B. Kahng was born in 1963, at San Diego, CA. He received the A.B. degree in applied mathematics/physics from Harvard College, Cambridge, MA, and the M.S. and Ph.D. degrees in computer science from the University of California, San Diego.

After 11 years on the University of California, Los Angeles, computer science faculty, most recently as Professor and Vice-Chair, he is now Professor in the University of California, San Diego, CSE and ECE departments since January 2001. He has published over 160 papers in the VLSI CAD literature, centering on physical layout and performance analysis. He currently leads the "C.A.D." theme in the MARCO Gigascale Silicon Research Center (http://vlsicad.cs.ucla.edu/GSRC/) and the U.S. Design Technology Working Group for the International Technology Roadmap for Semiconductors. His research interests include VLSI physical layout design and performance analysis, combinatorial and graph algorithms, and large-scale heuristic global optimization.

Dr. Kahng received the National Science Foundation Young Investigator, and the Design Automation Conference Best Paper awards. He was the founding General Chair of the ACM/IEEE International Symposium on Physical Design, and since 1997, has defined the physical design roadmap for the SIA International Technology Roadmap for Semiconductors.

Andrew Kennings received the B.A.Sc., M.A.Sc., and Ph.D. degrees in electrical engineering from the University of Waterloo, Waterloo, Canada in 1992, 1994, and 1997, respectively.

He is currently a Staff Software Engineer at Cypress Semiconductor. His research interests include VLSI physical layout, analytical and combinatorial optimization algorithms, and PLD architectures.

Igor Markov received the undergraduate degree in mathematics from the Kiev University, Ukraine in 1993, and the M.A. degree in pure mathematics and the Ph.D. degree in computer science, both from the University of California, Los Angeles, in 1994, and 2000, respectively.

In 1995, he worked for Parametric Technology Corporation, Waltham, MA, on solid modeling CAD software. He is now an Assistant Professor of computer science and electrical engineering at the University of Michigan, Ann Arbor.

Dr. Markov received the best Ph.D. student award for 2000 from the Computer Science Department at University of California, Los Angeles.