[17] J. Cong and C.-K. Koh, "Simultaneous driver and wire sizing for performance and power optimization," in *Proc. Int. Conf. on Computer Aided Design*, pp. 206–212, Nov. 1994.

[18] J. Lillis, C. K. Cheng, and T. T. Y. Lin, "Optimal wire sizing and buffer insertion for low power and a generalized delay model," in *Proc. Int. Conf. on Computer Aided Design*, pp. 138–143, Nov. 1995.

[19] N. Menezes, R. Baldick, and L. T. Pileggi, "A sequential quadratic programming approach to concurrent gate and wire sizing," in *Proc. Int. Conf. on Computer Aided Design*, pp. 144–151, 1995.

[20] C. Chu and D. F. Wong, "A new approach to simultaneous buffer insertion and wire sizing," in *Proc. Int. Conf. on Computer Aided Design*, pp. 614–621, 1997.

[21] Semiconductor Industry Association, *National Technology Roadmap for Semiconductors*, 1994.

[22] J. K. Ousterhout, "Switch-level delay models for digital MOS VLSI," in *Proc. Design Automation Conf*, pp. 542–548, 1984.

[23] K. Nabors and J. White, "Fastcap: A multipole accelerated 3-d capacitance extraction program," in *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1447–1459, Nov. 1991.

[24] A. R. Conn, P. K. Coulman, R. A. Haring, G. L. Morrill, and C. Visweswariah, "Optimization of custom MOS circuits by transistor sizing," in *Proc. Int. Conf. on Computer Aided Design*, pp. 174–180, 1996.

[25] J. Cong, L. He, C. Koh, and Z. Pan, "Global interconnect sizing and spacing with consideration of coupling capacitance," in *Proc. Int. Conf. on Computer Aided Design*, 1997.

[26] J. G. Ecker, "Geometric programming: Methods, computations and applications," *SIAM Review*, vol. 22, pp. 338–362, July 1980.

[27] J. Rubinstein, P. Penfield, Jr., and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-2, pp. 202–211, July 1983.

[28] J. Cong, L. He, A. B. Kahng, D. Noice, S. N., and S. H.-C. Yen, "Analysis and justification of a simple, practical 2 1/2-d capacitance extraction methodology," in *Proc. Design Automation Conf*, pp. 627–632, 1997.

[29] C. Chien, P. Yang, E. Cohen, R. Jain, and H. Samueli, "A 12.7Mchip/s all-digital BPSK direct sequence spread-spectrum IF transceiver in 1.2μm CMOS," in *Proc. IEEE Int. Solid-State Circuits Conf.*, pp. 30–31, 1994.

problems are solved in the context of simultaneous device and wire sizing optimization for deep submicron designs. Experiments show that our LR-based optimization algorithm is very effective and extremely efficient. Up to 16.5% delay reduction is observed when compared with previous work based on the simple device model [1], and up to 31% delay reduction and 100x speedup is observed when compared the global interconnect sizing and spacing work [2]. We believe that our general CH-posynomial formulation and LR-based algorithm can also be applied to other optimization problems in the CAD field.

References

[1] J. Cong and L. He, "An efficient approach to simultaneous transistor and interconnect sizing," in *Proc. Int. Conf. on Computer-Aided Design*, pp. 181–186, Nov. 1996.

[2] J. Cong, L. He, C. Koh, and Z. Pan, "Global interconnect sizing and spacing with consideration of coupling capacitance," in *Proc. Int. Conf. on Computer Aided Design*, 1997.

[3] J. Cong, L. He, K. K.Y., C. Koh, and Z. Pan, "Interconnect design for deep submicron ICs," in *Proc. Int. Conf. on Computer Aided Design*, 1997.

[4] J. Cong, L. He, C.-K. Koh, and P. H. Madden, "Performance optimization of VLSI interconnect layout," *Integration, the VLSI Journal*, vol. 21, pp. 1–94, 1996.

[5] J. P. Fishburn and A. E. Dunlop, "TILOS: A posynomial programming approach to transistor sizing," in *Proc. Int. Conf. on Computer Aided Design*, pp. 326–328, 1985.

[6] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex optimization," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 1621–1634, Nov. 1993.

[7] M. Berkelaar and J. Jess, "Gate sizing in MOS digital circuits with linear programming," in *Proc. European Design Automation Conf.*, pp. 217–221, 1990.

[8] W. Chuang, S. S. Sapatnekar, and I. N. Hajj, "Timing and area optimization for standard-cell VLSI circuit design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, pp. 308–320, Mar. 1995.

[9] G. Chen, H. Onodera, and K. Tamaru, "An iterative gate sizing approach with accurate delay evaluation," in *Proc. Int. Conf. on Computer Aided Design*, pp. 422–427, 1995.

[10] J. Cong, K. S. Leung, and D. Zhou, "Performance-driven interconnect design based on distributed RC delay model," in *Proc. Design Automation Conf*, pp. 606–611, 1993.

[11] J. Cong and K. S. Leung, "Optimal wiresizing under the distributed Elmore delay model," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, pp. 321–336, Mar. 1995.

[12] S. S. Sapatnekar, "RC interconnect optimization under the Elmore delay model," in *Proc. Design Automation Conf*, pp. 387–391, 1994.

[13] N. Menezes, S. Pullela, F. Dartu, and L. T. Pillage, "RC interconnect synthesis — a moment fitting appraoch," in *Proc. Int. Conf. on Computer Aided Design*, pp. 418–425, 1994.

[14] T. Xue and E. S. Kuh, "Post routing performance optimization via multi-link insertion and non-uniform wiresizing," in *Proc. Int. Conf. on Computer Aided Design*, pp. 575–580, 1995.

[15] J. Cong and L. He, "Optimal wiresizing for interconnects with multiple sources," in *Proc. Int. Conf. on Computer Aided Design*, pp. 568–574, Nov. 1995.

[16] C. P. Chen, Y. W. Chang, and D. F. Wong, "Fast performance-driven optimization for buffered clock trees based on Lagrangian relaxation," in *Proc. Design Automation Conf*, pp. 405–408, 1996.

extension of the GISS algorithm to more general capacitance model is beneficial. However, this extension to the general capacitance model is not optimal, and our LR-based optimization algorithm always achieves the best solution. Even compared with the GISS/VAF solution, the delay reduction may be up to 31%. More significantly, the LR-based algorithm is 100x faster than the GISS algorithm. Furthermore, it is worthwhile to mention that our LR-based optimization algorithm and implementation is capable to consider the device sizing problem and the wire sizing and spacing problem for multi-source nets, whereas the GISS algorithm is applicable to single-source nets without device sizing.

4.3 Discussions

Our experiments show that the identical LR-tight lower and upper bounds are often achieved for most wire segments and transistors, which immediately lead to optimal or near-optimal solutions. Therefore, in our experiments, we simply use the LR-tight lower bound as the final solution.

We noticed that under the table-based model for device, there were a little bit more devices and wires have non-identical lower and upper bounds. However, the table-based device model still gives more delay reduction when compared with the simple device model, even we simply treat the LR-tight lower bound as the final solution.

Meanwhile, identical lower and upper bounds were obtained for almost all wire segments in the experiment using table-based capacitance model. In other words, in terms of the "convergence" of lower and upper bounds, there was no significant difference between the table-based 2D capacitance model with consideration of coupling capacitance, and the simple capacitance model which only uses constant unit-area capacitance and fringe capacitance.

When the LR-tight lower and upper bounds do not meet, the gap between them is very small, often just of one discrete width in our experiments, and the percent of non-identical lower and upper bounds is also very small, thus bounded-enumeration can be carried out in reasonable time and the optimal solution can be achieved efficiently in practice. Moreover, the dynamic-programming method [18] can be applied within each DC-Connected-Component to find the local optimal solution between lower and upper bounds. It is worthwhile to mention that an LR-tight lower or upper bound has *zero* sensitivity, thus the sensitivity based method can not be used to obtain a solution better than an LR-tight lower or upper bound.

5 Conclusions

In this paper, we formulated a new class of optimization problem, named the general CH-posynomial program, which is more general than the simple and bounded-variation CH-posynomial programs in [1]. We revealed the general dominance property so that an efficient and unified algorithm based on the local refinement (LR) operation can be used to optimize the simple, bounded-variation and general CH-posynomial programs. We applied the LR-based optimization algorithm to solve the device sizing problem using accurate table-based model, and the wire sizing and spacing problem with consideration of coupling between multiple nets. Both

Pitch-spacing	Average Delay (ns)				Run Time (s)	
	MIN	GISS/FAF	GISS/VAF	LR-based	GISS/VAF	LR-based
2×	1.579	0.842(-47%)	0.842(-47%)	0.840(-47%)	317	2.3
3×	1.390	0.830(-40%)	0.579(-58%)	0.575(-59%)	321	4.7
4×	1.333	0.662(-50%)	0.511(-62%)	0.462(-65%)	349	4.6
5×	1.300	0.606(-53%)	0.414(-58%)	0.386(-70%)	932	9.7
6×	1.278	0.560(-56%)	0.479(-63%)	0.331(-74%)	2119	13.5

Table 4: Comparison of different sizing algorithm when sizing 16-bit buses. MIN is the minimum wire width (and thus maximum spacing) solution; GISS/FAF is GISS algorithm under the constraint of constant c_0 and c_1; GISS/VAF is GISS algorithm without considering the constraint of constant c_0 and c_1; LR-based is our LR-based algorithm presented in this paper.

We also observe that all other formulations lead to solutions using more transistor area than the *sgws/simple* formulation. However, wire areas are close under different formulations. Since the total area is dominated by the wire area, the increasing in term of the total area is reasonable. Not shown in this table is that solutions under all formulations have similar power dissipation. Although the *stis/table* formulation have the longest runtime, the maximum runtime is just 3.17 seconds. Not as expected, the *stis/step* formulation may take less time than the *sgws/step* formulation, even through both are very efficiently. Given these experiment results, the simultaneous transistor and wire sizing formulation using table-based model (*stis/table*) is recommended for high-performance designs.

4.2 Wire sizing and spacing for multiple nets

In the following experiment, we perform optimal wire sizing and spacing for a 16-bit parallel bus structure. Each bit is a $1cm$ line driven by a driver of 119 Ω. The sink capacitance is 12.0fF. An asymmetric wire sizing will be found for each $500\mu m$-long wire segment. We assume that these lines are initial equally spaced, and consider initial pitch-spacing from $2x$ to $6x$ of the minimum pitch space.

In this experiment, the minimum wire width W_{min} is $0.22\mu m$, and the minimum edge-to-edge spacing S_{min} is $0.33\mu m$. Therefore, the minimum pitch-spacing is $0.55\mu m$. The allowable wire are from 0.22 to 1.1 μm, with the incremental step of 0.11 μm. The area, fringe and coupling capacitances are obtained by the 2D look-up table [28] based on 3D extraction results via FastCap. We compare our LR-based optimization algorithm with the following sizing algorithms: minimum wire sizing (MIN) and GISS algorithm [2]. The GISS algorithm is based on the effective-fringe property to compute lower and upper bounds using bottom-up programming technique. Since the property assumes that the unit-area capacitance and fringe capacitance are constants, we denote it as GISS/FAF. It may be extended to use variable unit-area capacitance and fringe capacitance obtained directly from the table-based 2D capacitance model. We denote this case as GISS/VAF. Based on delays via HSPICE simulation in Table 4, even the MIN solution gives the largest spacing to reduce the coupling capacitance, it is still the worst solution. Compared with the MIN solution, the GISS/FAF solution reduces the maximum delay by from 47% to 56%. More delay reduction is achieved by the GISS/VAF solution, therefore,

net	maximum delay (ns)				runtime (s)			
	sgws/simple	sgws/table	stis/simple	stis/table	sgws/simple	sgws/table	stis/simple	stis/table
dclk	1.159	1.007(-6.4%)	1.132(-2.3%)	0.961 (-15.1%)	1.18	2.32	0.88	3.17
line	0.821	0.818(-0.4%)	0.751(-8.6%)	0.694(-16.5%)	0.72	0.58	0.55	1.22

net	average transistor size (μm)				average wire width (μm)			
	sgws/simple	sgws/table	stis/simple	stis/table	sgws/simple	sgws/table	stis/simple	stis/table
dclk	5.39	13.0(+242%)	17.2(+219%)	21.6(300%)	2.50	2.78(11.2%)	2.69(+7.6%)	2.82(12.8%)
line	108	112(+3.6%)	126(+16.7%)	125(15.7%)	4.98	4.99(+0.2%)	5.05(+1.4%)	5.11(+2.6%)

Table 3: Buffer and wire sizing for buffered trees under a $0.18\mu m$ process.

4 Experiments and Discussions

We have implemented our LR-based bound computation algorithm in C++ programming language, and test our algorithm under the $0.18\mu m$ technology given in [21]. We generate device delay tables using HSPICE, and generate capacitance tables using 3D capacitance tool FasctCap. The wire sheet-resistance is 0.0638 $\Omega/\square$ according to [21].

4.1 Simultaneous buffer and wire sizing

Our first set of experiments perform simultaneous buffer and wire sizing for two nets. One is a $2cm$ line with 5 buffers. These buffers are inserted for delay minimization. The other is a buffered tree, which is the *dclk* net in a spread spectrum IF transceiver chip design [29]. Its total wire length is 41518.2 μm, and there are 117 drivers and 37 buffers. The original design is in 1.2 μm 2-layer metal SCMOS technology, but in our experiment, we use parameters based on our 0.18 μm technology.

We compare sizing solutions under different sizing formulations (simultaneous gate and wire sizing (*sgws*) versus simultaneous transistor and wire sizing (*stis*) and different transistor models (simple model, i.e., r_0 is constant, versus table-based model). There are four combinations, namely, *sgws/simple*, *sgws/table*, *stis/simple* and *stis/table*. We assume the fixed ratio between p- and n- transistors in an inverter for gate sizing formulation is simply 1.0. For both nets, we find the optimal wire width for each 10 μm-long wire, and assume that allowable transistor sizes are multiples of $0.18\mu m$ between $0.18\mu m$ and $96\mu m$ and allowable wire widths are multiples of $0.56\mu m$ between $0.56\mu m$ and 5.6 μm. For simplicity, we assume constant unit-area capacitance c_0 and effective-fringe capacitance c_1, based on c_0 and c_1 values when the interested wire is $1.12\mu m$ wide and neighboring wires are $1.12\mu m$ away. This setting is suitable for the global routing stage, similar to our study in [3].

We achieved identical lower and upper bounds for most devices and wire segments for all formulations. Therefore, we simply use the LR-tight lower bounds as the final solution. According to HSPICE simulation in Table 3, when compared with the *sgws/simple* formulation, *sgws/simple*, *stis/simple* and *stis/table* formulations reduce the maximum delay by up to 6.4%, 8.6%, 16.5%, respectively. In addition to the fact that there always exist cases where the *stis/table* formulation outperforms other formulations by a large percent, the *stis/table* formulation also consistently achieves the best results. Note that the *stis* formulation perform closely to the *sgws* formulation for net *dclk*, due to the fact that the delay of net *dclk* is strongly dominated by the interconnect delay.

Similarly, a pass of upper bound computation starts with the current upper bound. The only difference is that we perform the LR operation for a wire E_i by using $c_0^{min}(i)$ and $c_1^{max}(i)$ (instead of $c_0(i)$ and $c_1(i)$) for E_i, and using $c_0^{max}(j)$ and $c_1^{max}(j)$ (instead of $c_0(j)$ and $c_0(j)$) for any edge E_j other than E_i. It is an ELR operation to obtain an upper bound for E_i.

Since the LR operation only computes the total width x_i for E_i, we need to map x_i into $(x_i^{\uparrow}, x_i^{\downarrow})$ to achieve an asymmetric wire sizing solution. Physically, this mapping is equivalent to embed a wire with width x_i around the initial central line of E_i, therefore we also call $(x_i^{\uparrow}, x_i^{\downarrow})$ an embedding. We define the following *greedy embedding*:

Definition 7 *For a given wire segment E_i of fixed neighboring wires and a total width x_i, the greedy embedding $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow})$ is an embedding to minimize the total capacitance of E_i.*

One is easy to verify that the greedy embedding leads to minimal delay among all embedding for a given total wire width. Since we use discrete spacings, the greedy embedding may not be unique. We have proved:

Lemma 1 *For a wire segment E_i with fixed neighboring wires, let x_i and x_i' be two possible widths for E_i, and $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow})$ and $(\hat{x}_i'^{\uparrow}, \hat{x}_i'^{\downarrow})$ be correspondent greedy embeddings. If $x_i \geq x_i'$, then for any $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow})$, there exits $(\hat{x}_i'^{\uparrow}, \hat{x}_i'^{\downarrow})$ such that $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow}) \geq (\hat{x}_i'^{\uparrow}, \hat{x}_i'^{\downarrow})$; and for any $(\hat{x}_i'^{\uparrow}, \hat{x}_i'^{\downarrow})$, there exits $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow})$, again, $(\hat{x}_i^{\uparrow}, \hat{x}_i^{\downarrow}) \geq (\hat{x}_i'^{\uparrow}, \hat{x}_i^{\downarrow})$.*

We propose to perform the greedy embedding operation after each LR operation. Since the optimal solution $\mathbf{X}^*$ $(= (\mathbf{X}^{*\uparrow}, \mathbf{X}^{*\downarrow}))$ must be a greedy embedding, from a lower (or upper) bound of $\mathbf{X}^*$ computed by LR operations, we are able to obtain lower (or upper) bounds for both $\mathbf{X}^{*\uparrow}$ and $\mathbf{X}^{*\downarrow}$ (Lemma 1). Therefore, our LR-based algorithm augmented by the greedy embedding operation computes LR-tight lower and upper bounds which are optimal in such a sense that there is an optimal asymmetric wire sizing and spacing solution within the two bounds.

3.3.3 Extensions to simultaneous sizing of multiple neighboring nets

We extend the above LR-based optimization algorithm to consider simultaneous sizing of multiple neighboring nets in a greedy fashion. For any wire segment E_i, we fix widths and locations of its neighboring wires, compute capacitance efficients $c_0^{min}(i)$, $c_0^{max}(i)$, $c_1^{min}(i)$ and $c_1^{max}(i)$, then perform LR operation. If minimum and maximum allowable widths are given for each wire segment, according to Theorem 2, *no* ELR operation will miss the optimal width for any wire segment. Therefore, with respect to the symmetric wire sizing formulation, the greedy extension lead to lower and upper bounds between which exists a *global* optimal wire sizing and spacing solution for multiple simultaneously sized neighboring nets.

Again, we can use the greedy embedding after each LR operation for the asymmetric wire sizing formulation. Even though we do yet prove that the greedy embedding leads to global optimal solution, our experiments show that the greedy embedding achieves results much better than the symmetric wire sizing formulation.

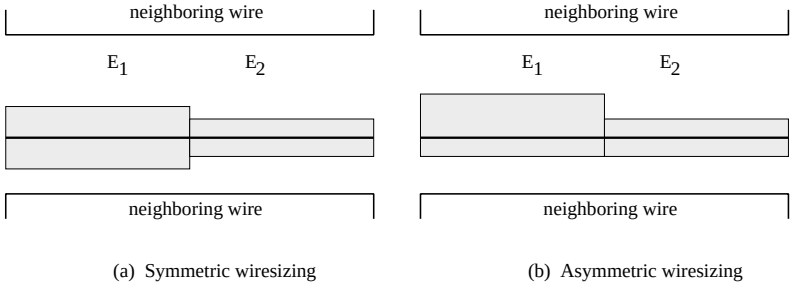

Figure 1: (a) Symmetric wire sizing, and (b) Asymmetric wire sizing.

formulation is capable to further reduce the interconnect delay.

In the asymmetric wire sizing formulation, the wire sizing and spacing for wire segment E_i is represented by $(x_i^{\uparrow}, x_i^{\downarrow})$, where $x_i^{\uparrow}$ is the width of the wire above (left to) the initial central-line when E_i is a horizontal (vertical) segment, and $x_i^{\downarrow}$ is the width of the wire in the other side of the initial central-line. In order to maintain the connectivity, we assume that $x_i^{\uparrow}$ and $x_i^{\downarrow}$ are at least $W_{min}/2$, where W_{min} is the minimum wire width set by the manufacture technology. We say that wire sizing and spacing solution $\mathbf{X}$ dominates another solution $\mathbf{X'}$ (denote as $\mathbf{X} \geq \mathbf{X'}$), if $(x_i^{\uparrow}, x_i^{\downarrow}) \geq (x'^{\uparrow}_i, x'^{\downarrow}_i)$ (i.e., $x_i^{\uparrow} \geq x'^{\uparrow}_i$ and $x_i^{\downarrow} \geq x'^{\downarrow}_i$) holds for any wire segment E_i.

Clearly, in Eqn. (8), coefficient functions (Eqn. 4–7) which depend on c_0 and c_1 are no longer constants, to minimize Eqn. (8) is not a simple CH-posynomial program. We proceed to solve it using LR-based optimization algorithm for the general CH-posynomial program. We will first consider the case that all neighboring wires are fixed with respect to interested nets, then extend to the general case where neighboring nets need to be sized simultaneously.

3.3.2 Sizing with fixed neighboring wires

Similar to the LR-based optimization algorithm for the STIS-D problem in Section 3.2, we start with initial lower bound $\mathbf{L}$ and upper bound $\mathbf{U}$, carry out interleave passes of lower and upper bound computations unitl lower and upper bounds meet or we can not further improve them.

For an edge E_i, we compute $c_0(i)$ for all valid wire widths (and spacings) within the current lower and upper bounds of wire sizing solution. We use $c_0^{min}(i)$ and $c_0^{max}(i)$ to denote the maximum and minimum values for $c_0(i)$. Similarily, $c_1^{min}(i)$ and $c_1^{max}(i)$ can be derived. With respect to these values, we start a pass of lower bound computation with the current lower bound. we use the LR operation for a transistor M_i since capacitance efficients of transistors are assumed constants in the STIS-C problem. We perform the LR operation for a wire E_i by using $c_0^{max}(i)$ and $c_1^{min}(i)$ (instead of $c_0(i)$ and $c_1(i)$) for E_i, and using $c_0^{min}(j)$ and $c_1^{min}(j)$ (instead of $c_0(j)$ and $c_0(j)$) for any edge E_j other than E_i. It is an ELR operation and we obtain a lower bound for E_i according to Theorem 2. We iterate through all transistors and wire segments until there is no improvement.

lower bound of the output t_t for M_i occurs when M_i is driven by a lower bound of the input t_t and is driving the upper bound of c_l. Symmetrically, $r_0^{max}(i)$ is determined using the upper bound of x_i, the upper bound of input t_t and the lower bound of c_l. As the lower and upper bound of sizing solution move closer during the LR-based optimization procedure, the range of r_0 is also narrowed. Our experiments in Section 4 show that identical lower and upper bounds are obtained for most wire segments and transistors.

In addition, we have proved that there exists an optimal sizing solution such that the optimal wire widths are monotonic within each wire segment. Therefore, we can use the bundled refinement operation [15] instead of LR operation for wire segment E_i. The bundled refinement operation is a speedup scheme for the LR operation, and is shown 100x time faster than the LR operation for the wiresizing problem.

3.3 STIS-C problem considering interconnect coupling

In addition to find the optimal sizes for transistors, the STIS-C problem is aimed to find optimal widths and spacings for all wire segments. We first present the capacitance model to consider interconnect coupling and the wire sizing and spacing (*WSS*) formulation. We then solve the STIS-C problem under this capacitance model and using the WSS formulaiton.

3.3.1 Capacitance model and WSS formulation

A table-based 2.5D capacitance model suitable for layout optimization was presented recently in [28]. The interconnect capacitance comprises the following components: area and fringe capacitances, coupling capacitance, and cross-over and cross-under capacitances. We consider only area, fringe and coupling capacitances (denoted as c_a, c_f and c_x) in this paper, since they are the major part of the interconnect capacitance. To be precise, we use a 2D capacitance model simplified from the original 2.5D model. In this 2D model, c_a, c_f and c_x for a wire depend only on the width of the wire and edge-to-edge spacings from the wire to its nearest neighboring wires on the same layer. Therefore, we first use 3D field solver to build tables for c_a, c_f and c_x under different width and spacing combinations. Then, table lookup is used during layout optimization to obtain c_a, c_f and c_x for given wire width and spacing, and c_0 and c_1 are derived based on $c_0 = c_a/w$ and $c_1 = c_f + c_x$ with w being the wire width.

We assume that an initial layout is given such that the *initial central-lines* and *initial pitch-spacings* defined by it remain unchanged during the optimization procedure. Since initial pitch-spacings remain unchanged during optimization, $c_0(i)$ and $c_1(i)$ only explicitly depends on width x_i for wire segment E_i, even through $c_a(i), c_f(i)$ and $c_x(i)$ are functions of width and edge-to-edge space of E_i in our 2D capacitance model.

We consider two wire sizing and spacing formulations introduced in [2]. One is the *symmetric* wire sizing formulation, where wires are always symmetric with respect to initial central-lines as illustrated in Figure 1(a). In contrast, the other is the *asymmetric* wire sizing formulation shown in Figure 1(b), where wires of same widths are asymmetric with respect to initial central-lines, and has smaller capacitance and delay. Given that neighboring wires are in general asymmetrically away from interested nets, the asymmetric wire sizing

3.2 STIS-D problem using accurate device model

The table-based model is used to accurately model a device. For each type of transistor, a number of SPICE simulations are carried out for different combinations of transistor sizes, input transition times and output loads. Then, values for the min-size effective resistance r_0 are pre-computed and stored in three-dimensional tables like Table 1. In addition to min-size resistance r_0, output transition times which are used to compute input transition times for the fanout are also pre-computed and stored in three-dimensional tables. Similar tables can be built for gates under both rising and falling inputs.

Given the fact that r_0 depends on the transistor size, input transition time and output load in the table-based model, and there is a large range for r_0 (see Table 1), r_0 is unlikely a function of a single sizing variable. It is nature to treat it as a function of the whole sizing solution $\mathbf{X}$. Since Eqn. (8) is a general CH-posynomial when r_0 is a function of $\mathbf{X}$, the STIS-D problem using the table-based device model is a general CH-posynomial program. Therefore, the LR-based optimization algorithm can be used to compute lower and upper bounds of the optimal solution to the STIS-D problem.

For a lower bound $\mathbf{X}$ of the optimal solution, since the min-size resistance $r_0(i)$ for each wire segment E_i is a constant, we perform the LR operation for E_i, and obtain a lower bound of the optimal size for E_i according to Theorem 1. Since $r_0(i)$ for transistor M_i is a function of sizing solution $\mathbf{X}$, we perform the LR operation for M_i by using $r_0^{min}(i)$ (instead of $r_0(i)$), and $r_0^{max}(j)$ (instead of $r_0(j)$) for any transistor M_j other than M_i, where $r_0^{min}(i)$ is the minimum possible value for $r_0(i)$ and $r_0^{max}(j)$ is the maximum possible value for $r_0(j)$. In other words, we perform an ELR operation for M_i and obtain a lower bound of the optimal size for M_i according to Theorem 2. We iteratively perform LR or ELR operations for all wire segements and transistors until we can not improve the lower bound. It is one *pass* of the lower bound computation.

For an upper bound $\mathbf{X}$ of the optimal solution, we perform the LR operation for a wire segment E_i and obtain an upper bound of the optimal size for E_i. We perform the ELR operation for M_i by using $r_0^{max}(i)$ (instead of $r_0(i)$), and $r_0^{min}(j)$ (instead of $r_0(j)$) for any transistor M_j other than M_i, where $r_0^{max}(i)$ is the maximum possible value for $r_0(i)$ and $r_0^{min}(j)$ is the minimum possible value for $r_0(j)$. We obtain an upper bound of the optimal size for M_i. We iteratively perform LR or ELR operations for all wire segements and transistors until we can not improve the upper bound. It is one pass of the upper bound computation.

Starting with the initial lower and upper bounds ($\mathbf{L}$ and $\mathbf{U}$), we carry out *interleave* passes of lower and upper bound computations until lower and upper bounds meet or we can not tighten them. We determine the minimum and maximum values for r_0 according to current lower and upper bounds. In our implementation, we assume that r_0 increases with respect to an increase of the transistor size and input transition time (the input t_t), but it decreases with respect to an increase of output load c_l. Therefore, $r_0^{min}(i)$ for M_i can be obtained by table lookup using the lower bound of size x_i, the lower bound of the input t_t and the upper bound of c_l. We use c_l under the current upper bound of the sizing solution as the upper bound of c_l. We set two initial lower and upper bounds for t_t, and update these two bounds during optimization procedure by assuming that the

In order to simultaneously minimize delays along *multiple critical paths* in a circuit, it is proposed to minimize the weighted delay $t(\mathbf{X})$ of all stages in the set of critical paths denoted as $\mathcal{P}$:

$$t(\mathbf{X}) = \sum_{P(N_s,N_t)\in\mathcal{P}} \lambda^{st} \cdot t(P(N_s,N_t),\mathbf{X}) \tag{3}$$

where the penalty λ^{st} indicates the criticality of stage $P(N_s,N_t)$. It can be provided either by the designer (after timing simulation) or computed iteratively using the Lagrangian-relaxation method as in [16] to minimize the maximum delay. Let

$$F_0(i,j) = \sum_{P(N_s,N_t)\in\mathcal{P}} \lambda^{st} \cdot f_0^{st}(i,j) \tag{4}$$

$$F_1(i,j) = \sum_{P(N_s,N_t)\in\mathcal{P}} \lambda^{st} \cdot f_1^{st}(i,j) \tag{5}$$

$$G(i) = \sum_{P(N_s,N_t)\in\mathcal{P}} \lambda^{st} \cdot g^{st}(i) \tag{6}$$

$$H_1(i) = \sum_{P(N_s,N_t)\in\mathcal{P}} \lambda^{st} \cdot h_1^{st}(i). \tag{7}$$

and eliminate those terms independent of $\mathbf{X}$, Eqn. (3) is re-written as

$$\begin{aligned} t(\mathbf{X}) = & \sum_{i,j} r_0(i)\cdot F_0(i,j)\cdot\frac{x_j}{x_i} + \sum_{i,j} r_0(i)\cdot F_1(i,j)\cdot\frac{1}{x_i} + \\ & \sum_{i} r_0(i)\cdot G(i)\cdot\frac{1}{x_i} + \sum_{i} r_0(i)\cdot H_1(i)\cdot\frac{1}{x_i} \end{aligned} \tag{8}$$

With respect to Eqn. (8), the following simultaneous transistor and interconnect sizing (STIS) problem is defined:

Formulation 1 *Given a circuit and the lower and upper bounds (denoted as* $\mathbf{L}$ *and* $\mathbf{U}$*) for the width of each transistor and wire, the simultaneous transistor and interconnect sizing (STIS) problem for delay minimization is to determine the discrete width for each transistor and wire (or equivalently, a sizing solution* $\mathbf{X}$*,* $\mathbf{L}\le\mathbf{X}\le\mathbf{U}$*) such that the weighted delay through multiple critical paths given by Eqn. (8) is minimized.*

Eqn. (8) is a simple CH-posynomial when all coefficients of sizing variables are constants under the simple model to treat r_0, c_0 and c_1 as constants. We call the STIS problem under this simple model as the *simple STIS* problem, which was solved in [1] based on the formulation of the simple CH-posynomial program. Since the simple model is not valid for DSM designs as shown in Section 1, in this paper, we study the STIS problem under an accurate device model where r_0 is no longer a constant and call it the *STIS-D* problem. We also study the STIS problem considering the interconnect coupling where neither c_0 nor c_1 is a constant and call it the *STIS-C* problem. For simplicity, we assume that c_0 and c_1 are constants for the STIS-D problem, and r_0 is a constant for the STIS-C problem. We will solve the two problems respectively in Sections 3.2 and 3.3 by posing them as the general CH-posynomial problem. The general STIS problem where all r_0, c_0 and c_1 are *not* constants can be solved by combining algorithms for the STIS-D and STIS-C problems.

Based on Theorem 2, if replacing LR-operation by ELR operation, we can still apply the LR-based algorithm to compute a set of lower and upper bounds in polynomial time. As we tighten the lower and upper bounds, range for the value of a coefficient is also narrowed. The algorithm guarantees that within the LR-tight lower and upper bounds, there exist an optimal solution to minimize the general CH-posynomial. We will use it to solve the simultaneous device and wire sizing problem using accurate device model and considering interconnect coupling.

3 Problem Formulation and Algorithm

3.1 Simultaneous transistor and interconnect sizing formulation

Since the continuous gate sizing formulation can be treated as a special case of the transistor sizing formulation, we use transistor sizing formulation in this paper. Similar to [1], our delay formulated is based on the delay for a stage. A stage is defined as a DC-connected path from a power supply (either the Vdd or the ground) to the gate node of a transistor, containing both transistors and wires. The delay of a stage $P(N_s, N_t)$ where N_s is its source and N_t is its sink can be written as Eqn. (2) according to the Elmore delay formulation in [27].

$$\begin{aligned} & t(P(N_s, N_t), \mathbf{X}) \\ = & \sum_{i,j} r_0(i) \cdot f_0^{st}(i,j) \cdot \frac{x_j}{x_i} + \sum_{i,j} r_0(i) \cdot f_1^{st}(i,j) \cdot \frac{1}{x_i} + \\ & \sum_{i} r_0(i) \cdot g^{st}(i) \cdot \frac{1}{x_i} + \sum_{i} r_0(i) \cdot h_0^{st}(i) + \sum_{i} r_0(i) \cdot h_1^{st}(i) \cdot \frac{1}{x_i} \end{aligned} \tag{2}$$

where $r_0(i)$ is the min-size resistance for a transistor M_i or a wire E_i, and x_i is its width. Coefficients $f_0^{st}, f_1^{st}, g^{st}, h_0^{st}$ and h_1^{st} are determined in the following:

$$f_0^{st}(i,j) = \begin{cases} c_{g0}(j) & \text{if } M_i/E_i \textit{ charges only the gate of } M_j \\ c_{d0}(j) & \text{if } M_i/E_i \textit{ charges only the drain of } M_j \\ c_{s0}(j) & \text{if } M_i/E_i \textit{ charges only the source of } M_j \\ (c_{d0}(j) + c_{s0}(j)) & \text{if } M_i/E_i \textit{ charges both the drain and the source of } M_j \\ c_0(j) & \text{if } M_i/E_i \textit{ charges } E_j \\ 0 & \text{otherwise} \end{cases}$$

$$f_1^{st}(i,j) = \begin{cases} c_1(j) & \text{if } M_i/E_i \textit{ charges } E_j \\ cg_1(j) & \text{if } M_i/E_i \textit{ charges } M_j \\ 0 & \text{otherwise} \end{cases}$$

$$g^{st}(i) = \begin{cases} \sum c^l & \text{if } M_i/E_i \textit{ charges loading capacitance } c^l \\ 0 & \text{otherwise} \end{cases}$$

$$h_0^{st}(i) = \begin{cases} \frac{c_0(i)}{2} & \text{if } E_i \in P(N_s, N_t) \\ 0 & \text{otherwise} \end{cases}$$

$$h_1^{st}(i) = \begin{cases} \frac{c_1(i)}{2} & \text{if } E_i \in P(N_s, N_t) \\ 0 & \text{otherwise} \end{cases}$$

where $c_{g0}(j), c_{s0}(j)$ and $c_{d0}(j)$ are respectively unit-area capacitances for the gate, source and drain of transistor M_j, and $c_{g1}(j), c_{s1}(j)$ and $c_{d1}(j)$ are correspondent fringe capacitances. In addition, $c_0(j)$ and $c_1(j)$ are unit-area and effective-fringe capacitances for wire E_j.

The dominance property leads to a *LR-based* algorithm to compute a set of lower and upper bounds of the optimal solution to minimize a CH-posynomial. Beginning with a solution $\mathbf{X} = \mathbf{L}$, we traverse every x_i in any order to perform a LR operation on it. Because $\mathbf{X}$ is dominated by $\mathbf{X}^*$, its local refinement is still dominated by $\mathbf{X}^*$. This process is repeated and $\mathbf{X}$ becomes increasingly closer to but always dominated by $\mathbf{X}^*$. It produces a set of lower bounds of $\mathbf{X}^*$ and is stopped when no improvement is achieved in the last round of traversal. Similarly, a set of upper bounds for $\mathbf{X}^*$ can be obtained by performing LR operations beginning with $\mathbf{X} = \mathbf{U}$. Let a lower or upper bound of $\mathbf{X}^*$ be LR-tight if it can not be tightened by any LR operation. The LR-based optimization algorithm obtains LR-tight lower and upper bounds in $O(r \cdot n)$ time, where r is the total number of the possible evaluations for all $x_i (i = \{1, \cdots, n\}) \in \mathbf{X}$. It is optimal in the sense that there exists an optimal solution between LR-tight lower and upper bounds.

2.2 Theory of general CH-posynomial program

In this paper, we propose to remove any constraint for coefficients in a CH-posynomial Eqn. (1), which leads to the *general CH-posynomial*.

Definition 5 (General CH-posynomial) *Eqn. (1) is a general CH-posynomial, if coefficients are functions of vector* $\mathbf{X}$*, and for* $\mathbf{L} \leq \mathbf{X} \leq \mathbf{U}$*, the value of any coefficient is bounded, i.e., for any* p, q, i *and* j*,* $a_{pi}^{min} \leq a_{pi}(\mathbf{X}) \leq a_{pi}^{max}$*, and* $b_{qj}^{min} \leq b_{qj}(\mathbf{X}) \leq b_{qj}^{max}$*).*

We extend our definition of local refinement operation to consider a general CH-posynomial program, the optimization problem to minimize a general CH-posynomial.

Definition 6 (Extended Local Refinement Operation) *For any solution* $\mathbf{X}'$*, the extended local refinement operation with respect to any particular variable* x_i *and general CH-posynomial* $f(\mathbf{X})$ *is to minimize* $f(\mathbf{X})$ *only by evaluating* x_i *while keeping the value of any* $x'_j (j \neq i)$ *in* $\mathbf{X}'$ *and using the following coefficients:*
when $\mathbf{X}' \geq \mathbf{X}^*$*, for any* p *and* q*, we use* a_{pi}^{max} *instead of* $a_{pi}(\mathbf{X}')$ *for terms* $\frac{1}{x_i^p}$*,* a_{pj}^{min} *instead of* $a_{pj}(\mathbf{X}')$ *for terms* $\frac{1}{x_j^p}$ *with* $j \neq i$*,* b_{qi}^{min} *instead of* $b_{qi}(\mathbf{X}')$ *for terms* x_i^q*, and* b_{qj}^{max} *instead of* $b_{qj}(\mathbf{X}')$ *for terms* x_j^q *with* $j \neq i$*;*
when $\mathbf{X}' \leq \mathbf{X}^*$*, for any* p *and* q*, we use* a_{pi}^{min} *instead of* $a_{pi}(\mathbf{X}')$ *for terms* $\frac{1}{x_i^p}$*,* a_{pj}^{max} *instead of* $a_{pj}(\mathbf{X}')$ *for terms* $\frac{1}{x_j^p}$ *with* $j \neq i$*,* b_{qi}^{max} *instead of* $b_{qi}(\mathbf{X}')$ *for terms* x_i^q*, and* b_{qj}^{min} *instead of* $b_{qj}(\mathbf{X}')$ *for terms* x_j^q *with* $j \neq i$*.*

We say that the result solution is the *extended local refinement* of $\mathbf{X}'$ (with respect to x_i). Later on, we use *ELR* to denote the extended local refinement.

We have proved the following theorem:

Theorem 2 (General Dominance Property) *Let* $\mathbf{X}^*$ *be the optimal solution to minimize a general CH-posynomial* $f(\mathbf{X})$ *Eqn. (1). For any* $\mathbf{X}'$ *dominating* $\mathbf{X}^*$*, a local refinement of* $\mathbf{X}'$ *by the ELR operation still dominates* $\mathbf{X}^*$*; For any* $\mathbf{X}'$ *dominated by* $\mathbf{X}^*$*, a local refinement of* $\mathbf{X}'$ *by the ELR operation is still dominated by* $\mathbf{X}^*$*.*

Then, the *simple CH-posynomial* is defined:

Definition 1 (Simple CH-posynomial) *Eqn. (1) is a simple CH-posynomial if coefficients* $a_{pi}(\mathbf{X})$ *and* $b_{qj}(\mathbf{X})$ *are constants.*

Since exponents in the simple CH-posynomial must be natural numbers, the class of the simple CH-posynomial is a subset of the class of the posynomial [26].

Moreover, the *bounded-variation CH-posynomial* is defined:

Definition 2 (Bounded-Variation CH-posynomial) *Eqn. (1) is a bounded-variation CH-posynomial if coefficients satisfy the following conditions: (i) for any* p *and* i, $a_{pi}(\mathbf{X})$ *is a function depending only on* x_i. *With respect to an increase of* x_i, $a_{pi}(\mathbf{X})$ *monotonically increases for any* p, *but* $\frac{a_{pi}(\mathbf{X})}{x_i^p}$ *still monotonically decreases for any* $p \neq 0$. *(ii)for any* q *and* j, $b_{qj}(\mathbf{X})$ *is a function depending only on* x_j. *With respect to an increase of* x_j, $b_{qj}(\mathbf{X})$ *monotonically decrease for any* q, *but* $b_{qj}(\mathbf{X}) \cdot x_j^q$ *still monotonically increases for any* $q \neq 0$.

Note that the bounded-variation CH-posynomial is originally called the general CH-posynomial in [1]. In this paper, we rename it and use the name general CH-posynomial for a more general class of formulation defined in Definition 5 later on.

Clearly, a bounded-variation CH-posynomial is no longer a posynomial. Yet, following the notations in [26], a *CH-posynomial program* is defined as an optimization problem to minimize a CH-posynomial Eqn. (1), subject to $\mathbf{L} \leq \mathbf{X} \leq \mathbf{U}$ (i.e., $l_i \leq x_i \leq u_i$ for $i = 1, \cdots, n$). It may be a simple or bounded-variation CH-posynomial program with respect to a simple or bounded-variation CH-posynomial. The dominance property is revealed for the CH-posynomial program, based on following concepts:

Definition 3 (Dominance Relation) *For two vectors* $\mathbf{X}$ *and* $\mathbf{X}'$, *we say that* $\mathbf{X}$ *dominates* $\mathbf{X}'$ *(denoted by* $\mathbf{X} \geq \mathbf{X}'$*) if* $x_i \geq x'_i$ *for* $i = 1, \cdots, n$.

Definition 4 (Local Refinement Operation) *For a solution vector (or simply, solution)* $\mathbf{X}'$, *the local refinement operation with respect to any particular variable* x_i *and function* $f(\mathbf{X})$ *is to minimize* $f(\mathbf{X})$ *by only evaluating* x_i *while keeping the value of any* $x'_j (j \neq i)$ *in* $\mathbf{X}'$ *and using coefficients with respect to* $\mathbf{X}'$ *in case of a bounded-variation CH-posynomial.*

Such an operation is also called *LR* operation in short. The result solution is called the *local refinement* of $\mathbf{X}'$ (with respect to x_i).

Theorem 1 (Dominance Property) *Let* $f(\mathbf{X})$ *be a simple or bounded-variation CH-posynomial, and* $\mathbf{X}^*$ *an optimal solution to minimize* $f(\mathbf{X})$. *For any solution* $\mathbf{X}'$ *of* $f(\mathbf{X})$, *if* $\mathbf{X}'$ *dominating* $\mathbf{X}^*$, *a local refinement of* $\mathbf{X}'$ *still dominates* $\mathbf{X}^*$; *if* $\mathbf{X}'$ *dominated by* $\mathbf{X}^*$, *a local refinement of* $\mathbf{X}'$ *is still dominated by* $\mathbf{X}^*$.

seconds to optimize a 16-bit bus of 320 wire segments in [2].

In this paper, the simultaneous device and wire sizing problem with consideration of general device and capacitance models is studied in the context of DSM designs. One significant contribution is that our algorithm is capable to apply arbitrarily accurate models for device delay and wire capacitances (including coupling between neighboring wires), using table-lookup and/or high-order complex characteristic functions. Our implementation uses table-based models for both device delay and wire capacitances. Device tables are generated using HSPICE simulation, and capacitance tables are generated using FastCap for 3D extraction. These table entries are very accurate. Interpolation and extrapolation are used for data points not in the tables. These modeling techniques are widely used in industry for DSM designs. Experiments show that up to 16.5% delay reduction is observed when compared with previous work based on the simple device model [1], and up to 31% delay reduction is observed when compared the global interconnect sizing and spacing work [2]. Another significant contribution is that our algorithm remains *extremely* efficient. The aforementioned 16-bit bus is optimized at the same machine in only 13.5 seconds versus 2119 seconds in [2], and a routing tree of 154 drivers/buffers and 41518.2μm-long wires (each wire segment is 10μm long) is optimized in 3.17 seconds.

Our algorithm is based on a new class of optimization problem formulated in this paper, called the *general CH-posynomial program*, which is more general than the simple and bounded-variation CH-posynomial programs in [1]. We revealed the general dominance property under this formulation and developed an efficient and unified algorithm based on the local refinement (LR) operation to optimize the simple, bounded-variation and general CH-posynomial programs. We solved our simultaneous device and sizing problem using table-based models for both device delay and interconnect coupling by posing it as instances of the general CH-posynomial program. We believe that our general CH-posynomial formulation and LR-based algorithm can also be applied to other optimization problems in the CAD field.

The remainder of this paper is organized as follows. In Section 2, we present the theory for the general CH-posynomial program and LR-based optimization algorithm. In Section 3, we apply the theory to solve the simultaneous transistor and interconnect sizing problem using table-based device model and considering coupling between neighboring wires. We present experiment results in Section 4, and conclude the paper in Section 5.

2 Theory of CH-Posynomial Programs

2.1 Review of simple and bounded-variation CH-posynomial programs

In [1], the CH-posynomial (*Cong-He* posynomial) is defined as a function of positive vector $\mathbf{X} = \{x_i \mid i = 1, \cdots, n\}$ with the following form:

$$\begin{aligned} f(\mathbf{X}) &= \sum_{p=0}^{m}\sum_{q=0}^{m}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\left(\frac{a_{pi}(\mathbf{X})}{x_i^p}\right)\cdot\left(b_{qj}(\mathbf{X})\cdot x_j^q\right) \\ where \quad & a_{pi}(\mathbf{X}) \geq 0 \ and \ b_{qj}(\mathbf{X}) \geq 0 \end{aligned} \tag{1}$$

These assumptions have been shown unrealistic, especially for DSM designs. In Table 1, we computed r_0 via HSPICE simulation for an inverter under the rising input (i.e., r_0 of the n-transistor in the inverter) based on the representative $0.18\mu m$ technology in SIA roadmap [21] for two different sizes ($100x$ and $400x$). Different combinations of input transition times and output loads are used for measuring. As one can see, r_0 is clearly a function of size, input transition time and output load. Its value may differ by *a factor of 2* (note that a similar observation was made in [22] as well, with smaller variation of effective resistance in the technology generation then). Furthermore, c_0 and c_f are no longer sufficient to model a wire capacitance, due to increasing importance of coupling capacitance in DSM designs. We generated area capacitance (c_a), fringe capacitance (c_f) and coupling capacitance (c_x) for a wire in our $0.18\mu m$ technology with three different widths and spacings via a 3D solver FastCap [23]. As one can see in Table 2, when wires are placed closer and the wire aspect ratio increases in DSM designs, c_x between neighboring wires can be 83% of the total wire capacitance. Therefore, it is needed to lump c_f and c_x into the effective-fringe capacitance c_1. The capacitance for wire of width w and length l now is given by $c_0 \cdot w \cdot l + c_1 \cdot l$. Moreover, neither c_0 nor c_f is a constant. Their values may differ by 16.7% and 60.9%, respectively, for different widths and spacings in Table 2.

(width, spacing)	($0.22\mu m$, $0.33\mu m$)	($0.44\mu m$, $0.66\mu m$)	($0.88\mu m$,$1.32\mu m$)
$c_a(aF/\mu m)$	7.390	16.64	28.25
$c_f(aF/\mu m)$	26.94	29.90	43.35
$c_x(aF/\mu m)$	178.2	107.4	16.86
c_x/c_{total}	83%	60%	19%
$c_0(aF/\mu m^2)$	33.50	37.82	32.44
$c_1(aF/\mu m)$	205.1	137.3	60.2

Table 2: Area capacitance (c_a), fringe capacitance (c_f) and coupling capacitance (c_x) of a wire for different wire widths and edge-to-edge spacings. We derive unit-area capacitance c_0 according to $c_0 = c_a/w$ with w being the wire width, and effective-fringe capacitance c_1 according to $c_0 = c_f + c_x$.

We say that a device model is a *simple* model if it assumes that r_0 is a constant, and a capacitance model is a *simple* model if it assumes that both c_0 and c_f, or more general, both c_0 and c_1, are constants. In contrast, a device model is a *general* model if it uses a non-constant r_0 for higher accuracy, and a capacitance model is a *general* model if it considers coupling capacitance and does *not* assume that c_0 and c_f are constants. Simple device and capacitance models were used in most previous work, except a few recent works where more accurate general models were used. In [19], a sequential quadratic programming method is used to solve the simultaneous gate sizing and wire sizing problem under both simple gate model and a voltage-ramp gate model (a general model). The latter model achieves better results but is 10x slower. The transistor sizing in [24] also considers a general device model based on sensitivity computation exploiting a circuit simulation. Logic patterns are needed since circuit simulation is used. Two very recent works [25, 2] consider coupling capacitance between neighboring wires. However, both assumes that c_0 and c_f are constants. Furthermore, none of these works consider general device and capacitance models simultaneously. In addition, runtime is often high in these works. For example, it took 5844 seconds to optimize a routing tree of 298 wire segments in [19], and 2119

1 Introduction

The interconnect delay has become the dominant factor in determining the circuit performance in deep submicron (*DSM*) designs [3]. Many optimization techniques have been proposed to reduce interconnect delay, including interconnect topology optimization, buffer insertion, and device and interconnect sizing (see [4] for a comprehensive survey). As part of our effort to develop a unified methodology and platform for interconnect optimization, we study the simultaneous device and interconnect sizing problem in the context of DSM designs.

Most previous works size device and interconnect separately. The device sizing problem includes both transistor and gate sizing formulations. The transistor sizing assigns the optimal size for each transistor independently [5, 6]. The gate sizing determines the optimal size for a gate [7, 8, 9]. The interconnect sizing problem, also called the wiresizing problem, was first introduced in [10, 11] to determine the optimal width for each wire segment in interconnects and polynomial-time optimal algorithms were developed. Later on, alternative wiresizing algorithms were proposed in [12, 13, 14, 15, 16]. Recently, several studies consider the simultaneous device and interconnect sizing problem. The simultaneous gate and wire sizing formulation is used in [17, 18, 19, 20], whereas the simultaneous transistor and interconnect sizing (*STIS*) formulation is used in [1].

size = 100x								
	n-transistor				p-transistor			
c_l / t_t	0.05ns	0.1ns	0.2ns	0.3ns	0.05ns	0.1ns	0.2ns	0.3ns
0.225pF	12200	13370	19180	20210	17200	19920	24550	26040
0.425pF	8135	9719	12500	14880	17180	17190	18820	20730
0.825pF	8124	8665	10250	11680	17090	17150	17290	18130
1.625pF	8114	8170	8707	9458	16140	17140	17150	17200
3.225pF	7578	8137	8251	8572	14710	16940	17100	17120
size = 400x								
	n-transistor				p-transistor			
c_l / t_t	0.05ns	0.1ns	0.2ns	0.3ns	0.05ns	0.1ns	0.2ns	0.3ns
0.501pF	12200	15550	19150	30390	18200	19970	27030	32150
0.901pF	11560	13360	17440	20150	17340	19590	24560	26050
1.701pF	8463	9688	12470	14870	17070	17420	18790	20700
3.301pF	7725	8812	10420	12140	17030	16780	17440	18490
4.901pF	7554	8480	10010	11180	16090	17020	17060	17670

Table 1: Min-size resistance r_0 for a n-transistor of different sizes, input transition times (t_t) and output loads (c_l).

However, most previous work used over simplified models for devices and interconnects, which are not capable of modeling many DSM issues. For example, a gate of size d is modeled by an effective resistor $r_d = r_0/d$, where r_0 is the effective resistance of the min-size gate, and is assumed *independent* of the size, input waveform slope, and output load of the gate. Moreover, the capacitance for wire of width w and length l is given by $c_0 \cdot w \cdot l + c_f \cdot l$, where c_0 and c_f are unit-area capacitance and fringe capacitance for the wire. Both are assumed to be *constants*.

UCLA Computer Science Technical Report 970034
An Efficient Technique for Device and Interconnect Optimization in Deep Submicron Designs *

Jason Cong and Lei He
Department of Computer Science
University of California, Los Angeles, CA 90095

Abstract

In this paper, we formulated a new class of optimization problem, named the general CH-posynomial program, which is more general than the simple and bounded-variation CH-posynomial programs in [1]. We revealed the general dominance property so that an efficient and unified algorithm based on the local refinement (LR) operation can be used to optimize the simple, bounded-variation and general CH-posynomial programs. We applied the LR-based optimization algorithm to solve the device sizing problem using accurate table-based model, and the wire sizing and spacing problem with consideration of coupling between multiple nets. Both problems are solved in the context of simultaneous device and wire sizing optimization for deep submicron designs. Experiments show that our LR-based optimization algorithm is very effective and extremely efficient. Up to 16.5% delay reduction is observed when compared with previous work based on the simple device model [1], and up to 31% delay reduction and 100x speedup is observed when compared the global interconnect sizing and spacing work [2]. We believe that our general CH-posynomial formulation and LR-based algorithm can also be applied to other optimization problems in the CAD field.

*This work is partially supported the NSF Young Investigator Award MIP-9357582 and grants from Cadence Design Systems and Intel Corporation under the California MICRO program.