3. *Глушань В.М., Лаврик П.В.* Исследование клиент-серверной модели распределенной САПР электронных схем // Известия ЮФУ. Технические науки. Тематический выпуск «Интеллектуальные САПР». – 2009. – № 4 (93). – С. 77-81.
4. *Глушань В.М., Иванько Р.В., Лаврик П.В., Рыбальченко М.В.* Сравнительный анализ эффективности распределенных САПР // Труды Международных научно-технических конференций «Интеллектуальные системы» (AIS'06) и «Интеллектуальные САПР» (CAD-2006). – М.: Физматлит, 2006. – Т. 2. – С. 33-39.
5. *Оре О.* Теория графов. – М.: Наука, Главная редакция физико-математической литературы, 1980. – 336 с.

**Глушань Валентин Михайлович**
Технологический институт федерального государственного образовательного учреждения высшего профессионального образования "Южный федеральный университет" в г. Таганроге.
E-mail: gluval07@rambler.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 88634371651.
Кафедра систем автоматизированного проектирования; профессор.

**Glushan Valentin Mihailovich**
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education "Southern Federal University".
E-mail: gluval07@rambler.ru.
44, Nekrasovsky, Taganrog, 347928, Russia.
Phone: 88634371651.
Department of Computer Aided Design; professor.

**Лаврик Павел Викторович**
Технологический институт федерального государственного образовательного учреждения высшего профессионального образования "Южный федеральный университет" в г. Таганроге.
E-mail: gluval07@rambler.ru.
347928, г. Таганрог, пер. Некрасовский, 44.
Тел.: 88634371651.
Кафедра систем автоматизированного проектирования; аспирант.

**Lavrik Pavel Viktorovich**
Taganrog Institute of Technology – Federal State-Owned Educational Establishment of Higher Vocational Education "Southern Federal University".
E-mail: gluval07@rambler.ru.
44, Nekrasovsky, Taganrog, 347928, Russia.
Phone: 88634371651.
Department of Computer Aided Design; postgraduate student.

УДК 519.687.1

**А.Ф. Мелик-Адамян**

## МНОГОКРИТЕРИАЛЬНАЯ ОПТИМИЗАЦИЯ СТАНДАРТНЫХ КМОП-СХЕМ В СУБМИКРОННЫХ ТЕХНОЛОГИЯХ

*С уменьшением физических размеров транзисторов, адаптация стандартных ячеек и небольших схем в маршруте проектирования СБИС на этапе физического проектирования, является одной из актуальных задач в САПР микроэлектроники. Часто достижение тех или иных характеристик СБИС легче произвести адаптацией библиотек ячеек, нежели перепроектированием СБИС. В статье предложен метод многокритериальной оптимиза-*

*ции стандартных ячеек, годный для использования в существующих маршрутах проектирования. Экспериментальные результаты показывают улучшение характеристик энергопотребления и оптимизации площади до 15 %.*

*САПР микроэлектроники; генетические алгоритмы; многокритериальная оптимизация.*

**A.F. Melik-Adamyan**

## MULTIOBJECTIVE OPTIMIZATION OF STANDARD CELLS FOR CMOS NANO-SCALE VLSI DESIGNS

*With shrinking CMOS technology, the accurate trade-off between delay, static power consumption and yield of a digital circuit is becoming the most important factor while designing a functionally reliable and low power circuit . Gate sizing has emerged as one of the efficient way to achieve the goal in post-layout step of design flow. In the past single-objective optimization techniques have been used to optimize the timing variation, power or yield whereas on the other hand multi-objective optimization techniques can provide a more promising approach to design the circuit. We propose an algorithm based on multi-objective optimization technique called Non-dominated Sorting Genetic Algorithm. Algorithm overcomes the disadvantages of traditional optimization techniques.*

*Electronic design automation; genetic algorithm; multiobjective optimization.*

**Введение.** Продолжающейся развитие технологии производства полупроводниковой продукции, с технологической точки зрения, связано в первую очередь с уменьшением физических размеров транзисторов и уменьшением порогового напряжения. Эти факторы позволяют интегрировать на одной подложке несколько сот миллионов транзисторов. С уменьшением геометрических размеров транзисторов снижается площадь кристалла, уменьшаются паразитные ёмкости, улучшается быстродействие и снижается энергопотребление СБИС. За последние 30 лет длина затвора МОП-транзистора уменьшилась в 250 раз (с 10 мкм в начале 70-х годов до 40 нм в наши дни).

Однако такое уменьшение повлекло за собой появление новых проблем, которые не были важными в старых технологиях. Например, статическое энергопотребление, связанное с подпороговыми токами утечек. Начиная с технологии 90 нм и ниже, на подпороговые утечки приходиться до 50% от общего объема электропотребления для приложений, работающих на мобильных устройствах [1,2]. Другой, не менее важной проблемой является обеспечение уровня выхода годных (yield), повлекшее за собой создание новой дисциплины проектирования топологии с учетом технологических процессов (Design for Manufacturing) [9,10,12].

Одним из основных методов, отличающихся простотой, которую можно использовать в обоих случаях, является варьирование длин транзисторов. В работах [10-12] рассматриваются разные варианты этого метода для задач минимизации площади, энергопотребления и т.д. В основном, этот метод рассматривается при решении задач минимизации задержек при ограничениях на площадь. Оптимизация уровня выхода годных тоже в основном ставится в этих терминах [6,5]. Но большинство работ концентрируются на поиске компромисса (trade-off) между площадью и задержками. Однако в настоящее время энергопотребление или уровень выхода годных играют не меньшую роль в поисках компромиссов. Традиционная схема оптимизации выглядит следующим образом:

$$\begin{gathered} f(x) \rightarrow min \\ \text{при} \\ g(x) \leq g_{max} \\ h(x) < h_{max}, \end{gathered} \qquad (1)$$

где $g_{max}$, $h_{max}$ – ограничения. Оптимизационная задача с использованием только одного критерия (1), имеет несколько недостатков. Первое и самое главное: решение однокритериальной задачи оптимизации будет зависеть от ограничений, накладываемых на другие параметры, которые определяются инженером. Таким образом, если эти ограничения были выбраны неверно, то решение не будет иметь смысла или не даст максимального результата. Например, если мы определили штрафные очки при увеличении $g(x)$ и $h(x)$, и, если границы вариации целевых функций очень узкие, то оптимизация использует штрафные очки по максимуму. И может быть существуют другие решения, с меньшими значениями ограничений, которые мы не получим. Второе, большинство используемых техник оптимизаций в промышленных САПР, основываются на традиционных однокритериальных методах оптимизации. При наличии нескольких критериев, они оптимизируют сначала по одному критерию, а потом по второму и т.д. Результат такой «последовательной» оптимизации существенно зависит от выбора порядка критериев, по которым идет оптимизация. При этом теряется сам смысл многокритериальной оптимизации, когда идет одновременная оптимизация по нескольким критериям. И третье, при использовании многокритериальной оптимизации, при отсутствии решения мы можем получить близкое к оптимальному решение, что с инженерной точки зрения может быть приемлемым. Традиционные техники в таких случаях не дадут результата.

Таким образом, необходим метод который:

- одновременно оптимизирует по нескольким критериям;
- максимально не требует вмешательства пользователя;
- даст решения, близкие к оптимальным, при отсутствии точного решения.

Применение генетических алгоритмов для задач САПР рассмотрены в [1-4]. В данной статье рассматривается техника многокритериальной оптимизации на основе комбинации ЛП-поиска [14] и оптимизации на основе генетического алгоритма недоминирующего упорядочивания [27-29]. Этот алгоритм пытается минимизировать все критерии одновременно. Алгоритм находит все Парето-оптимальные решения, если они существуют, или даст точки которые лучше исходных, но не попадают в множество оптимальных решений. Такой метод дает инженерам-проектировщикам гибкость при оптимизации и выборе компромиссов между решениями.

**Базовая модель размерной оптимизации.** Традиционная схема размерной оптимизации может быть представлена следующим образом [7]:

$$\begin{gathered} \text{дано } (G, O), \quad \text{найти } L; \quad min\ \sigma^2\ (d); \\ \text{при} \quad \mu(d) \le \mu_{\max}\ \pi(G) \le \pi_{\max}, \end{gathered} \tag{2}$$

где $G$ – множество всех транзисторов и $g_i \in G - O$ множество выходов ячейки, $O \subset G$; $L = \{l_1, l_2, \ldots, l_n\}$, где $l_i$ – длина транзистора $g_i$; $\mu(x)$ – математическое ожидание; $\sigma^2(x)$ дисперсия случайной величины $x$; $d_i$ – задержка $i$-того выхода ячейки, $i \in O$; $\pi(G)$ – площадь ячейки содержащее все транзисторы из $G$; $\pi_{\max}$ и $\mu_{\max}$ – максимальное допустимые площадь и математическое ожидание задержек выходов ячейки.

Таким образом, алгоритм решения задачи (2) должен решить нелинейную оптимизационную задачу и предоставить инженеру-проектировщику единственный вектор $L$. Ограничения определяют верхнюю грань для штрафных значений при оптимизации целевой функции, которая выступает в виде издержек на площадь и задержку.

**Общая схема процесса оптимизации.** При использовании оптимизации в стандартном маршруте проектирования, необходимо состыковать оптимизатор

с SPICE программой для точной оценки результатов. Общая схема оптимизации представлена на рис. 1.

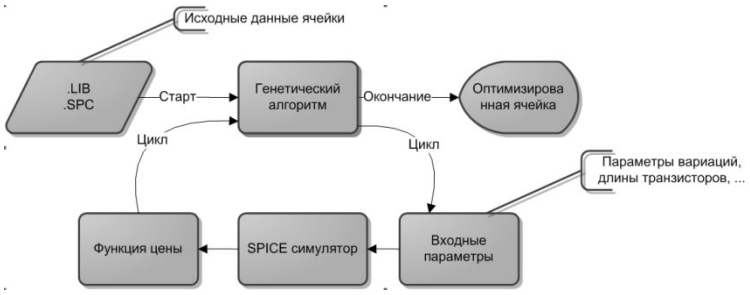


*Рис. 1. Схема оптимизации в маршруте проектирования*

Постановка нашей задачи для оптимизации при двух критериях и предоставлении нескольких близких к оптимальным, при отсутствии Парето-оптимальных решений выглядит следующим образом:

$$\begin{gathered} \text{найти } L;\ min\ \max(\sigma/\mu); \\ min\ \pi(G);\ min\ P(G); \\ \text{при } lj^{min} \le lj \le ljmax. \end{gathered} \tag{3}$$

Модель задержек соответствует модели, представленной в [23] вычисляемой с помощью метода STA.

А модель утечки *P* представлена далее.

**Модель тока утечки.** Большинство аналитических работ по моделированию подпороговых токов утечки используют модель BSIM4 [13],

$$I_{sub} = A \cdot \exp\left(\frac{V_{GS} - V_T - \gamma' V_{SB} + \eta V_{DS}}{n V_{TH}}\right)\left(1 - \exp\frac{-V_{DS}}{V_{TH}}\right), \tag{4}$$

где *VGS, VT, VSB, VDS, VTH* – напряжения, соответственно, определяющие затвор-исток, пороговое, исток-подложка, сток-исток, и термальное $V_{TH} = kT/q$, *n* – подпороговый коэффициент качения, $\gamma'$ – линейный коэффициент подложки, $\eta$ – коэффициент стоко-вызванный утечки через барьер (DIBL), и

$$A = \mu_0 C_{OX} \frac{W}{L} V_{TH}{}^2 \exp\left(\frac{-V_T}{\eta V_{TH}}\right) e^{1.8}, \tag{5}$$

где $\mu_0$ – коэффициент смещения нулевого напряжения, $C_{OX}$ — емкость окисла затвора, *W* — ψθπθνΰ, а *L* — длина транзистора.

Для определения тока утечки для ячейки, надо решить уравнение (1) для всех транзисторов, входящих в ячейку. Как мы видим, ток утечки, экспоненциально зависит от порогового напряжения, *VT,* температуры и коэффициентов $\gamma'$ и $\eta$.

Рассмотрим следующие ограничения:

- *VGS* = 0, нас будет интересовать только выключенное состояние транзистора;
- *VDS = VSB = VDD* – рассмотрим только один транзистор, без эффекта стекирования и взаимодействия нескольких транзисторов.

Как показано в [14], в таком случае, уравнение (1) принимает вид:

$$I_{sub} = I_0 \cdot \frac{W}{L} e^{-V_T'}, \tag{6}$$

а для $V_T'$, принимая во внимание ограничения и уравнения для порогового напряжения [14], мы будем использовать следующее приближение:

$$V_T' = V_T + V_{DD} \cdot \frac{-(L + c_1 L^2)}{c_2} + c_3, \tag{7}$$

где *c1*, *c2* и *c3* – корректирующие параметры. Минус перед множителем *VDD* отражает тот факт, что транзистор с более короткой длиной канала имеет больший ток утечки.

Полный ток утечки для ячейки получается простым суммированием значений для всех транзисторов:

$$I_{total} = \alpha \sum_d I_{sub}, \tag{8}$$

где α – коэффициент определяемый экспериментальным путем, отражающий стекирование и соотношение размеров транзисторов.

Это уравнение может быть использовано для всех типов транзисторов. Эксперименты по уточнению (рис. 2) модели и определении точности были проведены для коммерческой библиотеки 130 нм, для варианта с скоростными ячейками. Эксперименты были проведены на компьютере с процессором Core2 Duo 7300, с оперативной памятью 2 ГБ, под управлением ОС Linux. Все программы написаны на языке C++. Были проверены 231 ячейка с использованием модели (3) и коммерческого симулятора Synopsys HSIM. Было организовано 10000 прогонов симулятора.

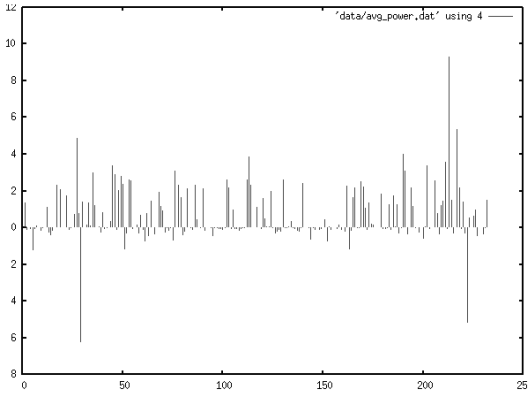


*Рис. 2. Отклонение эмпирической модели от BSIM4 (в %)*

В среднем программа для решения задачи (3) работает в 500 раз быстрее, чем симулятор HSIM, что позволяет его использовать для получения оценок с достаточной точностью для использования в схеме оптимизации.

**Модель площади.** Модель площади формируется простым суммированием площадей транзисторов умноженным на весовой коэффициент.

$$\pi(G) = \sum_{i=1}^{N} l_i w_i \alpha_i,$$

где $l_i$ – длина $i$-го транзистора, $i \in G$; $w_i$ – ширина $i$-го транзистора, $i \in G$; $\alpha_i$ – весовой $i$-го транзистора, $i \in G$, определяется его участием в трассировке и показывает, насколько изменение $i$-го транзистора влияет на общую площадь. Более детально о вычислении $\alpha_i$ можно узнать из работ [15,19,20].

**Генетический алгоритм.** Как видно из уравнений в разделах II-V и задачи (3) ставится задача нелинейной многокритериальной оптимизации. Это делает затруднительной применение традиционных схем оптимизации. В работах [27-29] показано, как применение генетических алгоритмов помогает в решении многокритериальных задач в САПР. Далее мы будем использовать схему генетического алгоритма предложенного в [29].

**Алгоритм 1.** ГА Алгоритм

```
pop = GenerateInitialPopulation
while generation ≤ max generation do
 rank = Ranking(pop)
       fitness = Fitness (pop, rank)
 for i = 1 to N step 2 do
  parent1 = Selection(pop, fitness)
  parent2 = Selection(pop, fitness)
       (child1, child2) = Crossover(parent1, parent2)
  newpopi = Mutation(child1)
  newpopi+1 = Mutation(child2)
 end for
 pop = newpop
 generation = generation + 1
end while
final rank = Ranking(pop)
Solutions = popi, ∀ i ∈ final ranki == 1
return Solutions
```

Алгоритм 1 представляет общую схему генетического алгоритма многокритериальной оптимизационной задачи, которая формулируется в общем виде следующим образом:

$$min\, f_i(\mathbf{x}) \quad \forall\, i \in \{1, M\}$$

$$\text{при } x'_j \le \mathbf{x_j} \le x''_j \quad \forall\, j \in \{1, P\}, \tag{9}$$

где $f_i$ – $i$-ая целевая функция, $i = 1, \ldots, M$ и $\mathbf{x} = \{x_1, \ldots, x_P\}$, где $x_j$ – $j$-ая варьируемая переменная и $x_j \in [x'_j, x''_j]$.

Рассмотрим шаги алгоритма более детально.

**Оператор `GenerateInitialPopulation`.**

Этот процесс генерирует массив *pop*, размерности $N \times P$, которая содержит $N$ (размер популяции) членов, каждая из которых является вектором длины $P$. Обычно популяция создается с помощью равномерно распределенных случайных величин, при условии, что каждая $x_j \in [x'_j, x''_j]$. Однако выбор начальной популяции может существенно ускорить сходимость процесса. Из равномерно распределенных случайных величин, мы для решения нашей задачи выбираем $ЛП_\tau$ последовательности Соболя [8]. Как показывают исследования, их основное свойство заключается в том, что они хорошо покрывают маломерные проекции многомерного куба. Например, проекции на все трёхмерные грани стомерного куба первых 128 членов $ЛП_\tau$-последовательности хорошо покрывают каждую трёхмерную грань, но плохо покрывают сам куб в силу его большой размерности. С практической точки

зрения, это означает, что $ЛП_\tau$-последовательности хорошо срабатывают для отыскания экстремумов функций, существенно зависящих от небольшого числа своих аргументов, т.е. , что функция

$$P(l_1, l_2, \ldots l_n) = P^*(l_{i1}, l_{i2}, \ldots l_{ik}) + g(l_1, l_2, \ldots l_n), \qquad (10)$$

где $k<n$, и $P^*>>g$. Также, на практике, в силу своей равномерной распределённости, они обеспечивают лучшую сходимость, чем псевдослучайные последовательности.

Как было показано в [11,12], интересующие нас функции энергопотребления и уровня выхода годных обладают свойством (10). Для такой функции решение задачи поиска экстремума близко к гиперплоскости. Это позволит ГА быстрее сходится и даже при отсутствии Парето-оптимальных решений, дать приемлемые для инженера-проектировщика решения. В [36,37] приведены алгоритмы быстрой генерации $ЛП_\tau$-последовательностей и их сравнение с другими равномерно распределенными последовательностями случайных величин.

**Оператор `Ranking`.**

Будем говорить, что решение $x^a$ доминирует над решением $x^b$, если имеют место следующие условия.

1. $x^a$ не хуже, чем $x^b$ для любой целевой функции $f_i(x^a) \geq f_i(x^b) \quad \forall i = 1,\ldots, M$.
2. $x^a$ строго лучше $x^b$, хотя бы для одной целевой функции $\exists i\, f_i(x^b) < f_i(x^a)$.

Ранжирование осуществляется с помощью следующего алгоритма.

**Алгоритм 2.** Ранжирование

```
χ = 1 { {xa, xb} ∈ pop}
do
   N = length(pop)
   for a = 1 to N do
        if ∃ b such that xb > xa then
    push(xa, dominated)
        end if
   end for
   for a = 1 to N do
        if xa !∈ dominated then
              rankxa ← χ
              remove(xa, pop)
        end if
   end for
   χ ← χ + 1
   clear dominated
while pop != {ϕ}
return rank
```

**Оператор `FitnessAssignment`.**

После того, как популяция размера $N$ ранжировано, значения приспособленности присваиваются каждому решению. В начале все решения с в одним фронте, т.е. с одинаковым рангом, получают одинаковое значение приспособленности $f_k$ и потом это значение совместно используется остальными решениями. Процедура совместного использования решения $x^a$ в фронте $k$ выглядит следующим образом:

1. Вычисляется эвклидово расстояние от другого решения $x^b$ в фронте $k$ как:

$$d_{ab} = \sqrt{\sum_{p=1}^{P} \left( \frac{x_p^a - x_p^b}{x_P'' - x_P'} \right)^2}.$$

2. Функция совместного использования вычисляется как в []:

$$S(d_{ab}) = \begin{cases} 1 - \left( \frac{d_{ab}}{\xi} \right)^2, \textit{если } d_{ab} \le \xi; \\ 0, \text{в противном случае.} \end{cases}$$

Далее вычисляется коэффициент фронта:

$$n_a = \sum_{b=1}^{p_k} S(d_{ab}),$$

где $p_k$ – количество решений в $k$-том фронте. Новое значение приспособленности вычисляется как

$$f_i' = \frac{f_k}{n_a}.$$

Эта процедура повторяется для всех фронтов с

$$f_{k+1} = \min(f_k) - \varepsilon,$$

где $\varepsilon$ – малое положительно число.

**Оператор `Selection`.**

После того как значения приспособленности присвоены всем решениям данной популяции, выбирается одно решение из пула. Вероятность выбора решения $x^a$ с значением приспособленности $f_a$ определяется следующей формулой [27]:

$$P(x^a) = \frac{f_a}{\sum_{j=1}^{N} f_j}.$$

**Оператор `Crossover`.**

Скрещивание является фундаментальным механизмом перегруппировки решений в генетических алгоритмах. После того как два родителя выбираются оператором Selection, эти два родителя скрещиваются с вероятностью 0.9 для порождения потомства. Операция Crossover может быть представлена следующим образом:

1. Вычислить $\beta q$ :

$$\beta_q = \begin{cases} (u\alpha)^{\frac{1}{\eta_c - 1}}, \text{если } u \le 1/\alpha; \\ \left( \frac{1}{2 - u\alpha} \right)^{\frac{1}{\eta_c - 1}}, \text{в противном случае,} \end{cases}$$

где $\eta c = 30$, $u$ – случайное число в диапазоне между 0 и 1, и $\alpha = 2 - \beta^{-(\eta c + 1)}$, где $\beta$ равно:

$$\beta = 1 + \frac{2}{x_i^b - x_i^a} \min[x_i^a - x_i', x_i^b - x_i''].$$

2. Потомки формируются следующим образом:

$$y_i^a = 0.5(x_i^a + x_i^b) - \beta_q \mid x_i^b - x_i^a \mid,$$
$$y_i^b = 0.5(x_i^a + x_i^b) + \beta_q \mid x_i^b - x_i^a \mid.$$

**Оператор `Mutation`.**

Операция мутации изменяет маленькую часть решений, приблизительно одну из 10,000. Мутация сама по себе не продвигает нас к решению, но она позволяет удостовериться, что популяция может быть подвергнута дальнейшей эволюции. После того, как два потомка генерируются оператором скрещивания, они мутируются с полиномиальной распределенной вероятностью [31]. Для каждого $y_i$ с $i \in \{1, P\}$ выполняются следующие действия с вероятностью $p_m$: вычисляется параметр $\delta_q$:

$$S(d_{ab}) = \begin{cases} \left[2u + (1-2u)(1-\delta)^{\eta_m+1}\right]^{\frac{1}{\eta_m+1}} - 1, \text{если } u \le 0.5; \\ 1 - \left[2(1-u) + 2(u-0.5)(1-\delta)^{\eta_m+1}\right]^{\frac{1}{\eta_m+1}}, \text{если } u > 0.5, \end{cases}$$

где $u$ – случайное число между 0 и 1, $\delta = \min[y_i^a - y_i', y_i^b - y_i'']$, а $\eta_m$ = 100+номер итерации.

Мутированная часть потомка вычисляется как: $z_i^a = y_i^a + \delta_q(y'' - y_i')$. Вероятность мутирования $p_m$ линейно варьируется между 1/P и 1.0.

Более детальное объяснение схем алгоритмов и процедур можно найти в [22-24].

**Экспериментальные результаты.** Алгоритм был опробован для нескольких библиотек стандартных схем, и подмножестве схем семейства ISCAS [32]. STA был сделан с помощью Berkley PTM [33] для 130 нм транзисторных моделей. Результаты работы алгоритма показаны на рис. 2, 3, из которых видно, что, несмотря на разное количество транзисторов, алгоритм предоставляет несколько решений для выбора инженером-проектировщиком. Все решения обозначенные '+' составляет Парето-оптимальные решения. Несколько решений из этого множества представлены в табл. 1. При использовании традиционной однокритериальной оптимизации, для отыскания max($\sigma/\mu$) с площадью, не превышающей 100 единиц, мы бы получили один набор длин транзисторов, который был бы в самом правом углу.

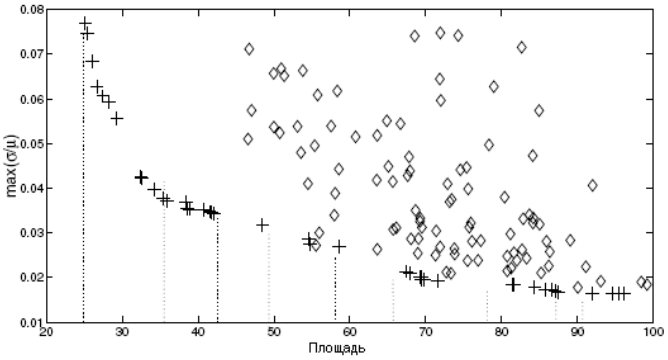


*Рис. 2. Крестиками обозначены результаты оптимизации площадь-задержка*

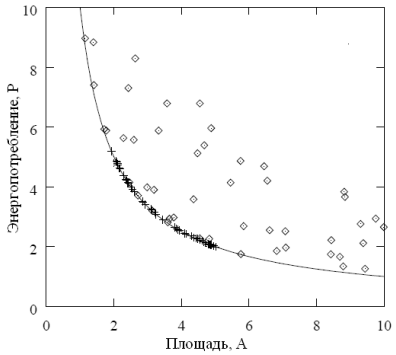


*Рис. 3. Крестиками обозначены результаты оптимизации площадь-энеропотребление*

Но, как видно из рисунка (см. рис. 3), существует целое множество решений, которые минимизируют целевую функцию, но при этом имеют меньший размер. Однокритериальная оптимизация не справляется с отысканием таких решений. С другой стороны, ГА позволяет выбрать любое решение из множества Парето-оптимальных решений, например, в левой части. Аналогичную ситуацию мы видим и с энергопотреблением.

Таблица 1

**Некоторые результаты оптимизации**

| Ячейка | Кол-во транзист. | Площадь | max($\sigma/\mu$) | Энергопотребл. | Время раб. с |
|---|---|---|---|---|---|
| 1 | 6 | 26.17 | 0.07 | 0.0914 | 3.76 |
| | | 33.51 | 0.04 | 0.0851 | |
| | | 53.82 | 0.02 | 0.0746 | |
| | | 72.61 | 0.01 | 0.0701 | |
| | | 95.98 | 0.01 | 0.0672 | |
| 2 | 160 | 768.61 | 0.05 | 6.1245 | 6.34 |
| | | 1053.99 | 0.03 | 5.1341 | |
| | | 2132.12 | 0.01 | 3.2515 | |
| | | 2912.14 | 0.01 | 3.0101 | |
| | | 3134.45 | 0.01 | 2.8642 | |

**Выводы.** В статье представлен новый подход к решению задачи многокритериальной оптимизации стандартных ячеек КМОП, вариацией длин транзисторов. Оптимизационный генетический алгоритм с использованием $ЛП_{\tau}$–поиска, позволяет получать несколько Парето-оптимальных решений, что позволяет внедрять предложенный метод в стандартный маршрут проектирования и использовать для очень широкого класса задач КМОП-схем.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Курейчик В.М., Лебедев Б.К., Лебедев О.К.* Поисковая адаптация. – М.: Физматлит, 2006.
2. *Гладков Л.А., Курейчик В.В., Курейчик В.М.* Генетические алгоритмы. – М.: Физматлит, 2006.
3. *Курейчик В.В., Курейчик В.М., Родзин С.И.* Концепция эволюционных вычислений, инспирированных природными системами // Известия ЮФУ. Технические науки. Тематический выпуск "Интеллектуальные САПР". – 2009. – № 4 (93). – С. 16-24.
4. *Лебедев Б.К.* Методы поисковой адаптации для решения оптимизационных задач // Новости искусственного интеллекта. – М., – 2000. – № 3. – С. 66-79.

5. *S.S. Sapatnekar, V.B. Rao, and P.M. Vaidya.* A convex optimization approach to transistor sizing for CMOS circuits // Proc. ACM/IEEE Int. Conf. Computer-Aided Design. – 1991. – P. 482-485,
6. *J.-M. Shyu, A. Sangiovanni-Vincentelli, J. Fishburn, and A. Dunlop.* Optimization-based transistor sizing // IEEE J. Solid-State Circuits, vol. 23. – P. 400-409, Apr. 1988.
7. *M.R. Berkelaar and J. A. Jess*, Gate sizing in MOS digital circuits with linear programming // Proc. European Design Automation Conf. – 1990. – P. 217-221.
8. *P.K. Chan.* Algorithms for library-specific sizing of combinational logic // Proc. ACM/IEEE Design Automation Conf. – 1990. – P. 353-356.
9. *S. Narendra, V. De, S. Borkar, D. Antoniadis, A. Chandrakasan.* Full-chip subthreshold leakage power prediction model for sub-0.18um CMOS // ISLPED, 2002.
10. *S. Mukhopadhyay,* A. Raychowdhury, *K. Roy.* Accurate estimate of total leakage current in scaled CMOS circuits based on compact current modeling // DAC 2003.
11. *R. Rao, A. Srivastava, D. Blaauw, D. Sylvester.* Statistical estimation of leakage current considering inter- and intra-die process variation // ISLPED, 2003.
12. *A. Srivastava, R. Bai, D. Blaauw, D. Sylvester.* Modeling and analysis of leakage power considering within-die process variations, ISLPED, 2002.
13. BSIM4 Reference Manual http://www-device.eecs.berkeley.edu/~bsim4/
14. *Соболь И.М., Статников Р.Б.* Выбор оптимальных параметров в задачах со многими критериями. – М.: Изд-во Дрофа, 2006.
15. *V. Oklobdzija.* Digital Design and Fabrication // CRC Press, 2008.
16. *C. Pigue.t* Low Power Electronics Design // CRC Press, 2007.
17. BSIM3 Reference Manual http://www-device.eecs.berkeley.edu/~bsim3/.
18. UC Berkeley Device Group. BSIM 4.2.1 MOSFET Model – User's Manual, 2004.
19. *S.H. Choi, B.C. Paul, K. Roy.* Novel sizing algorithm for yield improvement under process variation in nanometer technology // DAC, 2004.
20. *A.A.Ilumoka.* Optimal transistor sizing for CMOS circuits using modular artificial neural networks // Twenty-Ninth Southeastern Symposium on System Theory. – 1997.
21. *J. Singh, V. Nookala, Z.-Q. Luo, S. Sapatnekar.* Robust gate sizing by geometric programming // DAC. – 2005.
22. *V. Agarwal, J. Wang.* Yield-area optimizations of digital circuits using non-dominated sorting genetic algorithm (YOGA) // DAC. – 2006.
23. *M. Pan, C.C. N. Chu, H. Zhou.* Timing yield estimation using statistical static timing analysis // Intl Symposium on Circuits and Systems. – 2005.
24. *O. Neiroukh, X. Song.* Improving the process-variation tolerance of digital circuits using gate sizing and statistical techniques // Design Automation and Test in Europe. – 2005.
25. *M. Mani, A. Devgan, M. Orshansky.* An efficient algorithm for statistical minimization of total power under timing yield constraints // *DAC.* – 2005.
26. *C.M. Foncesa, P.J. Flemming.* Genetic algorithms for multi-objective optimization: Formulaton, discussion and generalization // In *5th Intl Conf on Genetic Algorithms.* – 1993. – P. 416-423.
27. *N. Srinivas, K. Deb.* Multi-objective function optimization using non-dominated sorting genetic algorithms // *Evolutionary Computatio.* – 1994.
28. *A.G. Cunha, P. Oliveira, J. A. Covas.* Use of genetic algorithms in multicriteria optimization to solve industrial problems // Proceedings of the Seventh International Conference on Genetic Algorithms. – 1997. – P. 682-688.
29. *K. Deb.* Non-linear goal programming using multi-objective genetic algorithms // Evolutionary Computation Journal. – 1994.
30. *J. Liou, K. Cheng, S. Kundu, A. Krstic.* Fast statistical timing analysis by probabilistic event propagation // DAC. – 2001. – P. 661-666.
31. *M. Hansen, H. Yalcin, J.P. Hayes.* Unveiling the iscas-85 benchmarks: A case study in reverse engineering // IEEE Design and Test. – 1999.
32. *P. Bratley, B. Lox, H. Niederreiter*. Implementation and Tests of Low-Discrepancy Sequences // ACM Transactions on Modeling and Computer Simulation. – Vol. 2, No. 3, July 1992. – P. 195-213.
33. *H. Hong, F. Hickernel.* Computational Investigations of Low-Disrepancy Sequences // ACM Transactions on Mathematical Software. – Vol. 23, No. 2, June 1997. – P. 266-294.

34. *H. Hong, F. Hickernel*. Algorithm 823: Implementing Scrambled Digital Sequences // ACM Transactions on Mathematical Software. – Vol. 29, No. 2, June 2003. – P. 95-109.
35. *S. Joe, F. Kuo*. Remark on Algorithm 659: Implementing Sobol's Quasirandom Sequence Generator // ACM Transactions on Mathematical Software. – Vol. 29, No. 1, March 2003. – P. 49-57.

**Мелик-Адамян Арег Фрикович**
Институт точной механики и вычислительной техники им. С.А. Лебедева РАН.
E-mail: areg@ipmce.ru.
119991, г. Москва, Ленинский проспект, д. 51.
Начальник лаборатории проектирования микроэлектроники и САПР.

**Melik-Adamyan Areg Frikovich**
The Lebedev Institute for Precise Mechanics and Computer Engineering RAS
E-mail: areg@ipmce.ru.
51 Leninskiy Prospekt, Moscow, 119991, Russia.
Head of semiconductor design services and EDA laboratory.