

Optimization in VLSI design, E2004

- Exercises for week four.

Jens Egeblad

September 30th, 2004

1 Exercise 1 - Sorry About Last Week! (15 min.)

In last week's exercises I had made two mistakes regarding Q1 and Q9.

For Q1 the constraints regarding x_2 and x_3 should have been:

$$\begin{aligned} x_2 &\geq n_{2l} \\ x_2 &\leq n_{2r} \\ x_3 &\geq n_{2l} \\ x_3 &\leq n_{2r} \end{aligned}, \tag{1}$$

I hope this did not confuse anyone too much.

For Q9 it should have said:

“... The force at the center of each bin (i, j) is evaluated by:

$$\mathbf{f}_{ij} = \frac{k}{2\pi} \sum_{i'=1}^m \sum_{j'=1}^m D_{i'j'} \frac{\mathbf{r}_{ij} - \mathbf{r}_{i'j'}}{|\mathbf{r}_{ij} - \mathbf{r}_{i'j'}|^2} w \cdot h,$$

where $D_{i'j'}$ is the density of bin (i, j) and r_{ij} is the center coordinate of bin (i, j) .

The Fast Fourier Transform can be used to evaluate

$$f(i, j) = \sum_{i'=1}^n \sum_{j'=1}^n g(i - i', j - j') \cdot h(i', j')$$

for all $i, j = 1, \dots, n$ in $O(n^2 \log n)$ time.”

Since solving the associated question is pretty hard otherwise I give you the following chance to score cheap points by answering Q9 from last week again:

Q0 : How can you use the Fast Fourier Transform to evaluate the force function $\mathbf{f}(i, j)$ for each bin (i, j) ? (write up the functions g and h)

... Sorry about the confusion - Jens.

2 Exercise 2 - Linearization Scheme (60 min.)

Assuming that all pins are placed in the center of the modules the star net-length of net is given as:

$$ST(n) = \sum_{m \in M(n)} (x_m - x_s)^2 + (y_m - y_s)^2,$$

where $M(n)$ are the modules connected by net n , x_m and y_m are coordinates of m , and $x_s = \frac{1}{|M(n)|} \sum_{m \in M(n)} x_m$ and $y_s = \frac{1}{|M(n)|} \sum_{m \in M(n)} y_m$. As shown in last week's exercises we can rewrite:

$$ST(n) = ST_x(n) + ST_y(n),$$

where e.g. $ST_x(n) = \sum_{m \in M(n)} (x_m - x_s)^2$.

Unfortunately the Star Net-Length is a poor estimate of the required wire-length because it is quadratic. Fortunately there is a way around that.

Q1 : Exploit the fact that $|x| = \frac{x^2}{|x|}$ to rewrite the x -part of $ST_x(n)$ as a linearized Star Net-Length $STL_x(n)$ which is linear in the distance between modules and the star-point (x_s, y_s) .

The linearized Star Net-Length has been used in real-life applications. However, there is a problem if we wish to use the linearized Star Net-Length in conjunction with e.g. Kraftwerk and Gordian. These algorithms use the Conjugate Gradient Method to find minimum for a *quadratic* function. Functions of the form:

$$f(\mathbf{x}) = \sum_{i=1}^n \frac{x_i^2}{|x_i|}$$

are not quadratic.

To solve this problem we simplify the problem. Instead of dividing by $|x_i|$ we could create

$$g(\mathbf{x}) = \frac{\sum_{i=1}^n x_i^2}{\sum_{i=1}^n |x_i|}$$

Of course in general $g(\mathbf{x}) \neq f(\mathbf{x})$ but it may serve as an estimate.

Q2 : Rewrite the $STL_x(n)$ based on this simplification.

If we calculate the denominator $(\sum_{i=1}^n |x'_i|)$ for specific x'_i , we may define:

$$h(\mathbf{x}) = \frac{1}{v(\mathbf{x}')} \sum_{i=1}^n x_i^2,$$

where $v(\mathbf{x}') = \sum_{i=1}^n |x'_i|$. Then $h(\mathbf{x})$ is a good estimate of $g(\mathbf{x})$ when $|x_i - x'_i| \leq \epsilon$ for all i .

When we minimize the x -component of the quadratic Star Net-Length we solve the following problem:

$$\text{minimize } Q(\mathbf{x}) = \sum_{n \in N} w(n) ST_x(n), \quad (2)$$

where N is set of all nets in the placement problem instance and $w(n)$ is some net-weight of n . This is done efficiently by the Conjugate Gradient Method (CGM). Define:

$$L(\mathbf{x}) = \sum_{n \in N} w(n) STL_x(n),$$

Q3 : Exploit the fact that you can apply similar assumptions to the $ST_x(n)$ $STL_x(n)$ as we did for $g(\mathbf{x})$ and $h(\mathbf{x})$ to define:

$$L'(\mathbf{x}) = \sum_{n \in N} \tilde{w}(n) ST(n),$$

Such that $L'(\mathbf{x})$ depends on some \mathbf{x}' and is close to $L(\mathbf{x})$ when \mathbf{x}' is close to \mathbf{x} , but $L'(\mathbf{x})$ is quadratic and can be minimized by CGM and has a “more” linear net-length. What are the new net-weights \tilde{w} ?

Help: \tilde{w} may depend on \mathbf{x}' for some \mathbf{x}' close to \mathbf{x} .

Of course $h(\mathbf{x})$ differs from $g(\mathbf{x})$ when \mathbf{x} differs too much from \mathbf{x}' . Assume you start with \mathbf{x}' and have a way to solve

$$\text{minimize } h(\mathbf{x}).$$

Q4 : Can you come up with way which iteratively refines $h(\mathbf{x})$ and $v(\mathbf{x}')$ such that you solve (2) while $h(\mathbf{x})$ comes arbitrarily close to $g(\mathbf{x})$?

Help: You may modify \mathbf{x}' and hence $v(\mathbf{x}')$.

3 Exercise 2 - Polynomial Reduction of “GLS for VLSI” neighborhood (45 min.)

The “GLS for VLSI - design” neighborhood consists of horizontal and vertical translations of modules (amongst other things). Let M be the set of modules. Say we wish to determine the optimal horizontal translation of a modules $m \in M$. Naively one would try all positions of m from left to right and choose the best one. If we only allow integer coordinates and say that the width of the chip is W then we only need to consider coordinates $\{0, \dots, W - 1\}$. Unfortunately

searching the neighborhood this way takes at least time $O(W)$ which is not polynomial in the number of modules $|M|$. In other words, if the chip is *very* large this method is *very* slow.

Instead we would like a way to find the optimal position in polynomial time of the number of modules. To simplify our task we disregard nets completely and we do not consider the augmented objective function but only the basic objective function. Enumerate the modules in M as $M = \{m_1, \dots, m_n\}$ where $n = |M|$. The basic objective function is now:

$$f(\mathbf{p}) = \sum_{i=1}^n \sum_{j=i+1}^n \text{overlap}(m_i, m_j, \mathbf{x}),$$

where $\text{overlap}(m_i, m_j, \mathbf{p})$ is the size of the area modules m_i and m_j overlap in the placement \mathbf{p} . In general let $x(r), y(r), w(r)$ and $h(r)$ be the x -coordinate, y -coordinate, width and height of rectangle r . Then

$$\text{overlap}(m, n, \mathbf{p}) = \text{overlap}_y(m, n, \mathbf{p}) \cdot \text{overlap}_x(m, n, \mathbf{p}),$$

where $\text{overlap}_x(m, n, \mathbf{p}) = \max(\min(x(m)+w(m), x(n)+w(n)) - \max(x(m), x(n)), 0)$ and $\text{overlap}_y(m, n, \mathbf{p}) = \max(\min(y(m)+h(m), y(n)+h(n)) - \max(y(m), y(n)), 0)$.

Now to further simplify the problem consider two rectangles a and b . Assume a overlaps with b in y -direction. I.e. “ $y(a) < y(b) + h(b)$ and $y(a) + h(a) > y(b)$ ”. Now consider $\text{overlap}(m, n, \mathbf{p})$ in placements where we vary $x(a)$ but fix $x(b)$.

- For a placement p_1 with sufficiently small $x(a)$ – such that a is left of b – we have $\text{overlap}(a, b, \mathbf{p}_1) = 0$.
- For a placement p_2 with $[x(a), x(a) + w(a)] \cap [x(b), x(b) + w(b)] \neq \emptyset$ – such that a and b overlap – we have $\text{overlap}(a, b, \mathbf{p}_2) > 0$.
- For a placement p_3 with sufficiently large $x(a)$ – such that a is right of b – we have $\text{overlap}(a, b, \mathbf{p}_3) = 0$.

Q6 : These observations are all very nice, but show that $f(x') = \text{overlap}(a, b, p(x'))$ is a piece-wise linear function in x' when the placement $p(x')$ sets $x(a) = x'$ and $x(b) = x_0$ by considering what happens to the overlap as a slides across b .

Q7 : Now given rectangles r_1, \dots, r_n and one rectangle a how fast can you find the \tilde{x} such that $g(\tilde{x})$ is global minimum for $g(x')$ where:

$$g(x') = \sum_{i=1}^n \text{overlap}(a, r_i, p(x'))$$

and $p(x')$ sets $x(a) = x'$ and $x(r_i) = x_i$ for some fixed x_i ? (i.e. how fast can you find the optimal translation of a)

Note: We have only considered a very simple version of the objective function used the GLS for VLSI article. However, it is possible to determine the optimal translation of a module even for the augmented objective function in polynomial time.

4 Exercise 4 - Mr. Baskerville's Heuristics (45 min) (Mandatory)

Mr. Baskerville – an English exchange student – has decided to attend the VLSI-course at DIKU. For the Global Placement exercises he has invented a new Global Placement Heuristic for last week's exercise 5.

His idea is to begin with the unconstrained linear programming formulation:

$$\text{minimize } \sum_{n \in N} BB(n), \quad (3)$$

where N is the set of nets and $BB(n)$ is the Bounding-Box net-length of net n . Of course, this generates solutions with massive overlap.

Inspired by Q3 and Q4 from last week's exercises he proposes to iteratively add constraints and integer variables that ensures that modules do not overlap. So his heuristic is as follows:

Algorithm 1: Global Placement Heuristic

```
Given Placement Problem with modules  $M$  and nets  $N$  ;
Let  $C$  be set of constraints. ;
Let  $I$  be set of integer variables ;
 $C = \emptyset$  ;
 $I = \emptyset$  ;
while Solution contains overlap do
  Solve: minimize  $\sum_{n \in N} BB(n)$  s.t. all constraints in  $C$  are
  uphold;
  if module  $m_i \in M$  and  $m_j \in M$  overlap then
    Add constraints and integer variables to  $C$  and  $I$  which en-
    force that  $m_i$  and  $m_j$  no longer overlap.
  end
end
```

Q8 : Mr. Baskerville claims that algorithm 1 is in fact an exact algorithm to solve the placement problem. Do you agree?

(Note: Both “No” and “Yes” may be valid so please argue why and why not one may agree.)

Mr. Baskerville's teacher does not really believe in this idea. He claims that the Global Placement heuristic is too slow to work in practice.

Q9 : Do you agree? Do you think the heuristic can handle 5, 10.000 or 1.000.000 modules within reasonable time (Guess and argue)?

Fortunately the following week Mr. Baskerville is asked to produce a Final Placement heuristic. Disappointed that his global placement heuristic was badly received he decides to produce a Local Search Heuristic instead.

This time instead of adding non-overlapping constraints his approach is different. His heuristic will iteratively select and extract k modules $M_r \subseteq M$ randomly and exchange their positions optimally. Assume that the *current* locations of the k modules are denoted (x'_i, y'_i) , $i = 1, \dots, k$, Mr. Baskerville is then attempting to solve the problem

$$\text{minimize } \sum_{n \in N} BB(n), \quad (4)$$

such that each of the k locations (x'_i, y'_i) , $i = 1, \dots, k$, is occupied by one and only one of the k modules from M_r and all other modules $M \setminus M_r$ are fixed at their current position.

Enumerate the modules in $M_r = \{m_1, \dots, m_k\}$ and let x_i be a variable that describes the x -coordinate of the current position of module $m_i \in M_r$. Let x_0 be the x -coordinate of one of the k -locations.

Q10 : Using integer or binary variables formulate constraints that ensures that one and only one of the k modules in M_r will have its x -coordinate equal to x_0 . Try to make the constraint linear using a large value M as part of the formulation.

The teacher is ecstatic about this idea. However, he is also hopelessly young and inexperienced and he tells Mr. Baskerville that it must be possible to come up with a polynomial time algorithm (in k) such that the k modules are placed optimally at the k different locations. Mr. Baskerville remembers that his teacher told him that even the VLSI-placement problem where all modules have equal size is NP-Hard. Mr. Baskerville knows little about VLSI-design but he does know a lot about a certain group of hard problems and he is pretty sure that a polynomial time algorithm for the optimal location of the k modules is very unlikely to exist.

Q11 : Which of the two are correct? (Guess and argue!)

5 Exercise 5 - The Sequence-Pair Representation Revisited (25 min)

It can be shown that no correct conversion of a placement of n rectangles can be done faster than sorting n distinct integer numbers.

Q12: Prove this lower bound!