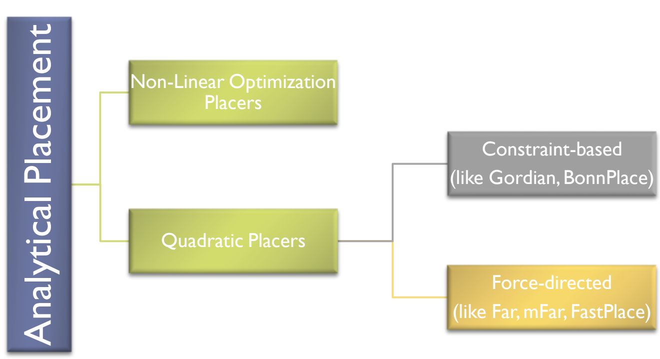
Introduction to KRAFTWERK 2

A Fast Force-Directed Quadratic Placer

by Giaourtas Mixalis & Vlachos Xanthos

Introduction

- Is an analytical placement techniques.
- Force-directed Quadratic placer
- The goal is to reduce the modules overlap into the core



Basic Idea

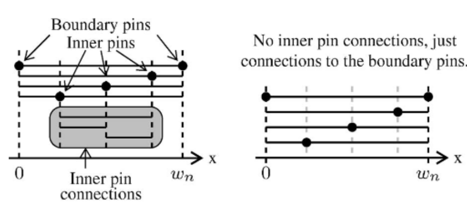
- Based on other, Force-Directed methods, Kraftwek2 uses the Bound2Bound net model, which:
  - Eliminates the approximation error, approaching the HPWL results.
  - It consists of:
    - Clique nets (connecting up to 5 pins)
    - Star nets (more than 5 pins)
  - Is used to determine the cost function, hence the Net Force.
- Two additional Forces are introduced to spread the modules
  - Move Force
  - Hold Force
- The target is to achieve a force equilibrium

Bound2Bound Net Model

- Linear complexity
- All kinds of net models consist of one or more two-pin connections
- For each connection, weight is determined by:

$$w_{x,pq}^{\mathrm{B2B}} = \begin{cases} 0, & \text{if pin } p \text{ and pin } q \text{ are inner pins} \\ \frac{2}{P-1}\frac{1}{|x_p^{\mathrm{pin}} - x_q^{\mathrm{pin}}|}, & \text{else.} \end{cases}$$

B2B approach on Clique nets



Generally about Net Model (1/2)

- Fundamental connection type: **two-pin connection p-q**
- Total Cost Function: $\Gamma = \frac{1}{2} \sum_{e=(p,q)\in\mathcal{E}} w_{x,pq}(x_p^{pin} - x_q^{pin})^2 + w_{y,pq}(y_p^{pin} - y_q^{pin})^2$
- Cost Function for each two-pin connection(x-dim): $\Gamma_{x,pq} = \frac{w_{x,pq}}{2}\left(x_{\pi(p)} - x_p^{off} - x_{\pi(q)} + x_q^{off}\right)^2$
  - *where* **π(ρ) = m**, function that relates p(pin() and m(module)
- Matrix notation of cost function: $\Gamma = \frac{1}{2}\mathbf{x}^T\mathbf{C_x}\mathbf{x} + \mathbf{x}^T\mathbf{d_x} + \frac{1}{2}\mathbf{y}^T\mathbf{C_y}\mathbf{y} + \mathbf{y}^T\mathbf{d_y} + const$
- Cx: represents the connectivity between M*(movable)* modules
- dx: reflects the connections of M and F*(fixed)* modules

Generally about Net Model (2/2)

- In x-dim the matrix notation: $\Gamma_x = \frac{1}{2}\mathbf{x}^T\mathbf{C_x}\mathbf{x} + \mathbf{x}^T\mathbf{d_x} + const.$
- By using the derivative: $\nabla_x\Gamma_x = \mathbf{C_x}\mathbf{x} + \mathbf{d_x}$

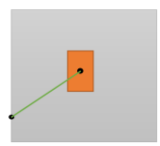
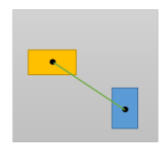
$$\begin{pmatrix} \vdots \\ \frac{\partial}{\partial x_i} \\ \vdots \\ \frac{\partial}{\partial x_j} \\ \vdots \end{pmatrix} \Gamma_x = \begin{pmatrix} \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & c_{x,ii} & \cdots & c_{x,ij} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \cdots & c_{x,ji} & \cdots & c_{x,jj} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \vdots \\ x_i \\ \vdots \\ x_j \\ \vdots \end{pmatrix} + \begin{pmatrix} \vdots \\ d_{x,i} \\ \vdots \\ d_{x,j} \\ \vdots \end{pmatrix}$$

- In order to obtain the minimum netlength: $\mathbf{C_x}\mathbf{x} + \mathbf{d_x} = \mathbf{0}$
- And finally i have found the x positions of my cells *(accordingly y)*

Net Force

$$\mathbf{F}_{\mathbf{x}}^{\mathrm{net}} = \nabla_x \Gamma_x = \mathbf{C}_{\mathbf{x}}\mathbf{x} + \mathbf{d}_{\mathbf{x}}.$$

- Net Force is the product of the nabla positions of the Movable cells with the Quadratic Cost Function for the x-dim. The y-dim is obtained similarly.
- It is transformed into a matrix-vector notation
- In reality, it is the force that keeps modules/cells to a specific position (pin), resulting in module overlap
- Acts as an elastic spring between pins-cells or cells-cells

Hold Force

- The hold force is

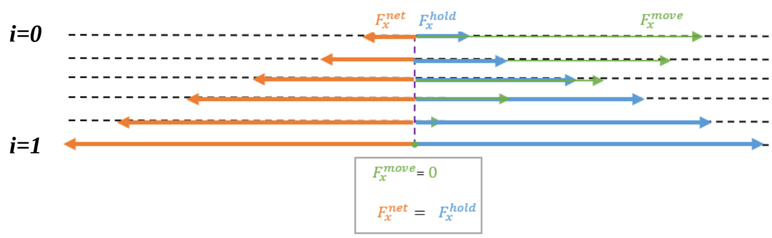
$$\mathbf{F}_{\mathbf{x}}^{\mathrm{hold}} = -(\mathbf{C}_{\mathbf{x}}\mathbf{x}' + \mathbf{d}_{\mathbf{x}})$$

- Throughout a cell movement F_x^{hold} eliminates the F_x^{net}.
- No force accumulation is necessary, because each iteration is decoupled from the previous one. Thus the placement algorithm can be restarted at any iteration

Move and Hold Force

- Move Force spreads the cells all over the chip area in order to reduce overlaps.

- Hold Force is a force to prevent moved cells from collapsing back to their previous position and always equal to the Net Force.



The Distribution Model (1/3)

- The distribution of the cells is modeled by a Demand-Supply System D.

$$\boldsymbol{D}(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{D}^{\boldsymbol{dem}}(\boldsymbol{x},\boldsymbol{y}) - \boldsymbol{D}^{\boldsymbol{sup}}\boldsymbol{sup}(\boldsymbol{x},\boldsymbol{y})$$

- The $D^{dem}(x,y)$ refers to the cells and the $D^{sup}(x,y)$ refers to the placement area (usually the core)

- Cells are moved away from high density regions(a lot of *DEMAND*) to low-density regions(remaining *SUPPLY*)

The Distribution Model (2/3)

- Due to the fact that $D^{dem} < D^{sup}$, the system **D** has to be balanced and adapted as it is presented, below:

$$\iint_{-\infty}^{+\infty} D^{dem}(x,y)dxdy = \iint_{-\infty}^{+\infty} D^{sup}(x,y)dxdy$$

- To formulate the demand, a rectangle function **R** is needed:

$$\boldsymbol{R(x,y,x_{ll},y_{ll},w,h)} = \begin{cases} \mathbf{1}, & \boldsymbol{if\ 0 \le x - x_{ll} \le w} \\ & \boldsymbol{\wedge\ 0 \le y - y_{ll} \le h} \\ \mathbf{0}, & \boldsymbol{elsewhere} \end{cases}$$

  - where,
    - $\boldsymbol{x, y}$ represent all the points inside the rectangle
    - $\boldsymbol{x_{ll}, y_{ll}}$ represent either the center or the left corner of a module/cell
    - $\boldsymbol{w, h}$ represent the width and height of the cell, accordingly.

The Distribution Model (3/3)

- For each cell $\boldsymbol{i}$, the demand

$$\boldsymbol{D_{cell}^{dem}(x,y) = d_{cell,i} * R(x,y,x'_i - \frac{w_i}{2}, y'_i - \frac{h_i}{2}, w_i, h_i)}$$

- The individual module density $d_{cell,i}$ is usually set to **1**.

- For each cell $\boldsymbol{i}$, the supply

$$\boldsymbol{D_{cell}^{sup}(x,y) = d_{sup} * R(x,y,x_{chip},y_{chip},w_{chip},h_{chip})}$$

- The supply density is:

$$\boldsymbol{d_{sup} = \sum_{i=1}^{M+F} \frac{(d_{cell,i} * A_m)}{A_{chip}}}$$

Electrostatic Potential Φ

- Then, **D** is interpreted as a charge distribution and creates an electrostatic potential φ, using Poisson's equation

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) * \varphi(x,y) = -D(x,y)$$

- Solving *(i.e. using MKL)* the equation we determine the target points

$$\dot{x}_i = x'_i - \frac{\partial}{\partial x}\varphi(x,y)\bigg|_{(x'_i, y'_i)}$$

previous location

- Gradients of the Potential φ are collected in a vector

$$\Phi = \left((\partial/\partial x)\varphi|_{(x'_1, y'_1)},\ (\partial/\partial x)\varphi|_{(x'_1, y'_1)},\ \dots,\ (\partial/\partial x)\varphi|_{(x'_M, y'_M)} \right)^T$$

Move Force

- The move force is

$$F_{x,i}^{move} = \dot{w}_i * (x_i - \dot{x}_i)$$

or

$$F_{x,i}^{move} = \dot{w}_i * (\underbrace{(x_i - x'_i)}_{\Delta x} + \Phi)$$

  - where,
    - $\dot{x}_i$ represents the target point, maximum new location of the cell i
    - x_i represents the new location of the cell i
    - x'_i represents the **old** location of the cell i
    - $\dot{w}_i$ represents the move force strength, by affecting the distance that the module is moved during each iteration of the algorithm.
- To ease the computation we transform the equation as $F_x^{move} = \dot{C}_x(x - \dot{x})$
  - where,
    - $\dot{w}_i$ are collected in a diagonal matrix $\dot{C} = diag(\dot{w})$, where $\dot{w}_i = \frac{A_{cell,i}}{A_{avg}} * \frac{1}{M}$

The Algorithm in "Graph Mode"

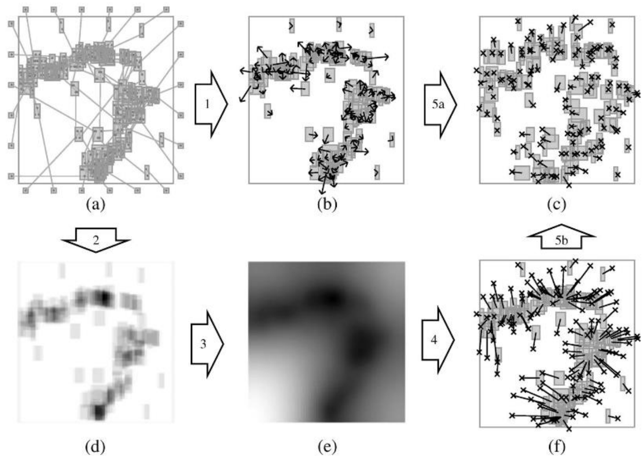


Fig. 6. Illustration of one placement iteration. The numbers in the big arrows represent the sequence of the steps executed in each placement iteration. (d), (e) Density plots, with white and black colors representing low and high densities, respectively. (a) Starting placement. (b) Hold force. (c) Resulting placement. (d) Supply and demand system D. (e) Potential Φ. (f) Target points and move force.

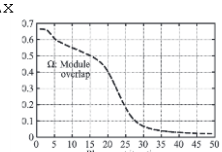
A Preview of the Algorithm

- The algorithm is **iterative** and is presented below

```
1.  Initial Placement

2.  While module Ω > 20%
            Create demand-and-supply system D(x,y)
            Calculate potential φ(x,y)
            Apply Bound2Bound net model
            For x-dim (analogously for y-dim)
                    Create C_x, Ċ_x, Φ_x
                    Solve (C_x + Ċ_x) * Δx = −Ċ_x * Φ_x
                    Update module position x by Δx
            Quality Control
    End While

3.  Final Placement
```



The Ω is the module overlap that is continuously decreasing over the placement iterations

Summarizing the Kraftwerk2 (1/2)

- The basic linear equation to be solved is

$$(C_x + \dot{C}_x) * \Delta \mathrm{x} = -\dot{C}_x * \Phi_x$$

  - C_x represents the cost function Γ
  - Φ_x vector represents the Demand-Supply system
  - $\dot{C}_x$ represents the weights of the move force

- In each iteration, we have a new position x for each module

- Both C_x and $\dot{C}_x$ are two square matrixes of dimension M*(number of movable objects) calculated also in each iteration*

- *Φx are the gradients of the potential φ, presented as a M size vector.*

Summarizing the Kraftwerk2 (2/2)

Let's clear things out a bit more:

$$\boldsymbol{F_x^{net} + F_x^{hold} + F_x^{move} = 0}$$

new position **old** position **target** position

$$F_x^{net} + F_x^{hold} = -F_x^{move}$$

$$(C_x * x + dx) + \left(-(C_x * x' + dx)\right) = -\dot{C}_x * (x - \dot{x})$$

$$C_x * x + \cancel{dx} - C_x * x' - \cancel{dx} = -\dot{C}_x * ((x - x') + \Phi_x)$$

$$C_x * x - C_x * x' = -\dot{C}_x * ((x - x') + \Phi_x)$$

$$C_x * (x - x') = -\dot{C}_x * ((x - x') + \Phi_x)$$

$$C_x * \Delta x = -\dot{C}_x * (\Delta x + \Phi_x)$$

$$C_x * \Delta x + \dot{C}_x * \Delta x = -\dot{C}_x * \Phi_x$$

$$\boldsymbol{(C_x + \dot{C}_x) * \Delta x = -\dot{C}_x * \Phi_x}$$

We solve this last form of the linear system, by determining the change in the module position Δx = x_new – x_old

How do the matrixes look like?

$$\dot{C}_x \begin{pmatrix} \frac{25}{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{23}{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{25}{6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{23}{6} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{23}{6} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{31}{6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{8}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{10}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{11}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{10}{3} \end{pmatrix}$$

MxM

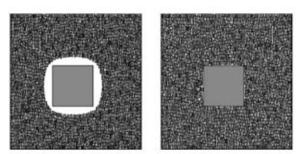
$$C_x \begin{pmatrix} \frac{25}{6} & -\frac{2}{3} & 0 & 0 & -\frac{7}{6} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ -\frac{2}{3} & \frac{23}{6} & -\frac{1}{2} & -\frac{1}{2} & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & \frac{25}{6} & -\frac{7}{6} & 0 & -\frac{2}{3} & -\frac{2}{3} & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & -\frac{7}{6} & \frac{23}{6} & 0 & 0 & 0 & 0 & 0 & -1 \\ -\frac{7}{6} & 0 & 0 & 0 & \frac{23}{6} & -\frac{1}{2} & 0 & -1 & 0 & 0 \\ -\frac{1}{2} & -1 & -\frac{2}{3} & 0 & -\frac{1}{2} & \frac{31}{6} & -\frac{2}{3} & -\frac{2}{3} & -\frac{2}{3} & 0 \\ 0 & 0 & -\frac{2}{3} & 0 & 0 & -\frac{2}{3} & \frac{8}{3} & 0 & -\frac{2}{3} & -\frac{2}{3} \\ 0 & 0 & 0 & 0 & -1 & -\frac{2}{3} & 0 & \frac{10}{3} & -\frac{2}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{2}{3} & -\frac{2}{3} & -\frac{2}{3} & \frac{11}{3} & -\frac{2}{3} \\ 0 & 0 & 0 & -1 & 0 & 0 & -\frac{2}{3} & 0 & -\frac{2}{3} & \frac{10}{3} \end{pmatrix}$$

MxM

Extras

- Avoid the "**Halos**", the free space around big cells

$$d_{cell,i} = \begin{cases} \sqrt{\frac{A_{\text{large}}}{A_m}}\,(1 - \text{td}) + \text{td} & A_m > A_{\text{large}} \quad \text{(i.e., large module)} \\ 1 & \text{otherwise} \end{cases}$$



- The quality of the placement adjusted/optimized by the way the weight factor is determined. In our case:

$$\dot{w}_i = \frac{A_{cell,i}}{A_{avg}} * \frac{1}{M}$$

- The D system defines the cells and the placement area(core) in our design. But it can be **extended** for other purposes like **routing supply and demand** so as to <u>optimize routability during placement.</u>

- And many others like the targeted or average movement of the cells.

Bibliography

- Kraftwerk2 – A Fast Force-Directed Quadratic Placement Approach Using an Accurate Net Model written *by Peter Spindler, Ulf Schlichtmann, Member, IEEE, and Frank M. Johannes*

- *Modern Circuit Placement – Best Practices and Results edited by Gi-Joon Nam, Jason Cong*

Questions

Any Questions?