# A Cool Title

A Cool Group Name

April 1, 2016

**Abstract**

Enter a short summary here of the entire group's work.

## Contents

**1 Introduction** **2**

**2 Work by Team 1** **3**

**3 Work by Team 2** **4**

**4 Work by Team 3** **5**

**5 Work by Team 4** **6**

**6 Work by Team 5** **7**

**7 Placement Problem** **8**

7.1 Introduction . . . . . 8
7.2 Problem Statement . . . . . 8
7.3 Problem Formulation . . . . . 8
7.4 Penalty Convex-Concave (Basic Algorithm) . . . . . 8
7.5 Method . . . . . 9
7.5.1 Initialization procedure . . . . . 9
7.5.2 Algorithmic Variation . . . . . 9
7.5.3 Problem Instance . . . . . 10
7.5.4 Computational details . . . . . 10
7.5.5 Results . . . . . 11
7.6 Other Non-CCP methods for Placement Problem . . . . . 11
7.7 Experimentation and Observations . . . . . 11
7.7.1 Effect on optimal value and number of iterations on varying $\mu$ . . . . . 12
7.7.2 Effect on optimal value and number of iterations on varying $\tau$ . . . . . 13

**8 Conclusion** **14**

**References** **14**

# 1 Introduction

## 2 Work by Team 1

## 3 Work by Team 2

# 4 Work by Team 3

## 5 Work by Team 4

## 6 Work by Team 5

# 7 Placement Problem

## 7.1 Introduction

Placement problem is to arrange non overlapping components in a way to minimize an objective function. The applications of this area include Circuit placing, floor-planning and placement of server clusters. In the following term paper we will aim at a specific problem,**Circuit Placement Problem**.

## 7.2 Problem Statement

Placement is an essential step in Electronic Design Automation, circuit placing problem involves minimization of total wire length i.e to minimize the sum of wire all the wires in the design. This not only helps in minimizing chip size and cost but also minimizes power and delay. With this notion in mind, following is the mathematical definition for circuit planning used in [2] :

- For components, consider $n$ square components.
- $X_i$, $X_i$ and $l_i$ for $i$ = *1,2...n* be the x position, y position and side length respectively of components $i$.
- The placement area is a rectangle with side length $b_x$ and $b_y$.
- A set of pairs $\epsilon$ representing the components we wish to be close.

## 7.3 Problem Formulation

Using the mathematical formulation of from [2] (mentioned above), the circuit placing problem is formulated as follows :

$$\text{Minimise } \sum_{(i,j)\in\epsilon} |x_i - x_j| + |y_i - y_j|$$
$$\text{subject to min } (\tfrac{l_i+l_j}{2} \text{ - } |x_i - x_j| \text{ , } \tfrac{l_i+l_j}{2} \text{ - } |y_i - y_j|) \leq 0,$$
$$i \text{ = 1,...,n-1, } j = i + 1, \ldots, n$$
$$|x_i| \leq (b_x - l_i)/2,\ i \text{ = 1,...,n } |y_i| \leq (b_y - l_i)/2,\ i \text{ = 1,...,n}$$

## 7.4 Penalty Convex-Concave (Basic Algorithm)

The basic convex-concave algorithm as defined in [2] is an extension to CCP algorithm. The normal gradient descent procedures forces to begin with a feasible point. However if slack variables are used, then initial point can be infeasible. In order to reach a feasible solutions slack variables are penalized. Initially a low penalty is put on the slack variables allowing for a greater deviation from the solution region.The pseudo code of the algorithm is given below : The stopping criterion would be reached if the change in objective is small i.e.
$(f_0(x_k) - g_0(x_k) + \tau_k \sum_{i=1}^{m} s_i^k) - (f_0(x_{k+1}) - g_0(x_{k+1}) + \tau_k \sum_{i=1}^{m} s_i^{(k+1)})) \leq \delta$,
where $s_i^k$ is the slack variable or $\tau_k = \tau_{max}$

The purpose of tau max is to ensure that the algorithm terminates even if no feasible region is reached. The theory of exact penalty functions[3] tells us that if tau is greater than largest optimal dual variable associated with the initial problem then solutions to our initial problem are solutions of the relaxed convexified problem. Thus if our tau max is greater than largest optimal dual

variable the value of tau max will have no impact on the solution. For nondifferentiable functions a subgradient has to be used whose choice will impact the performance of algorithm.

There are many different modifications which can be applied to constrained. The value of tau could be chosen on a per constraint basis prioritising some constraints. Then constraints that are purely convex could be enforced at all iterations without a slack variable such that giving us a feasible point at each iteration. Slack variables initially allowing for a violation to a convex constraint may allow us to reach a more favourable region of a non convex constraint.

**Algorithm** Penalty CCP

**given** an initial point $x_0, \tau_0 > 0, \tau_{max}$ , and $\mu > 1$.
k := 0

**repeat**

1: *Convexify.* Form $g_i(x; x_k) = g_i(x_k) + \nabla(x_k)^T(x - x_k)$ for $i$ = 0,...,$m$
2: *Solve.* Set the value of $x_{k+1}$ to a solution of
minimise $f_0(x) - g_0(x; x_k) + \tau_k \sum_{i=1}^{m} s_i$
subject to $f_i(x) - g_i(x; x_k) \leq s_i$, $i$ = 1, ..., $m$
3: *Update* $\tau.\tau_{k+1} := min(\mu\tau_k, \tau_{max})$.
4: *Update iteration.* $k := k + 1$.

**until** stopping criterion is satisfied.

## 7.5 Method

### 7.5.1 Initialization procedure

Two different types of initialization procedures can be used :

- From [3], One good initialization procedure is to use the embedding of the graph laplacian of the connections, this method involves eigenvectors corresponding to the two smallest eigenvalues of the Graph laplacian are used to initialize $x_i$ and $y_i$.
- Second initialization method is to uniformly initialize xi and yi in the appropriate range.

### 7.5.2 Algorithmic Variation

The problem is solved by using the Penalty convex-concave algorithm as mentioned in section- 4. The observations suggests that the two components are touching in the corner, the non- overlap constraint is non-differentiable, and therefore a subgradient must be chosen. Any linear separator between the vertical and horizontal separator is a valid subgradient. For breaking the ties, the vertical separators for even values of k and horizontal separator for odd values of k is chosen as default tiebreaker.

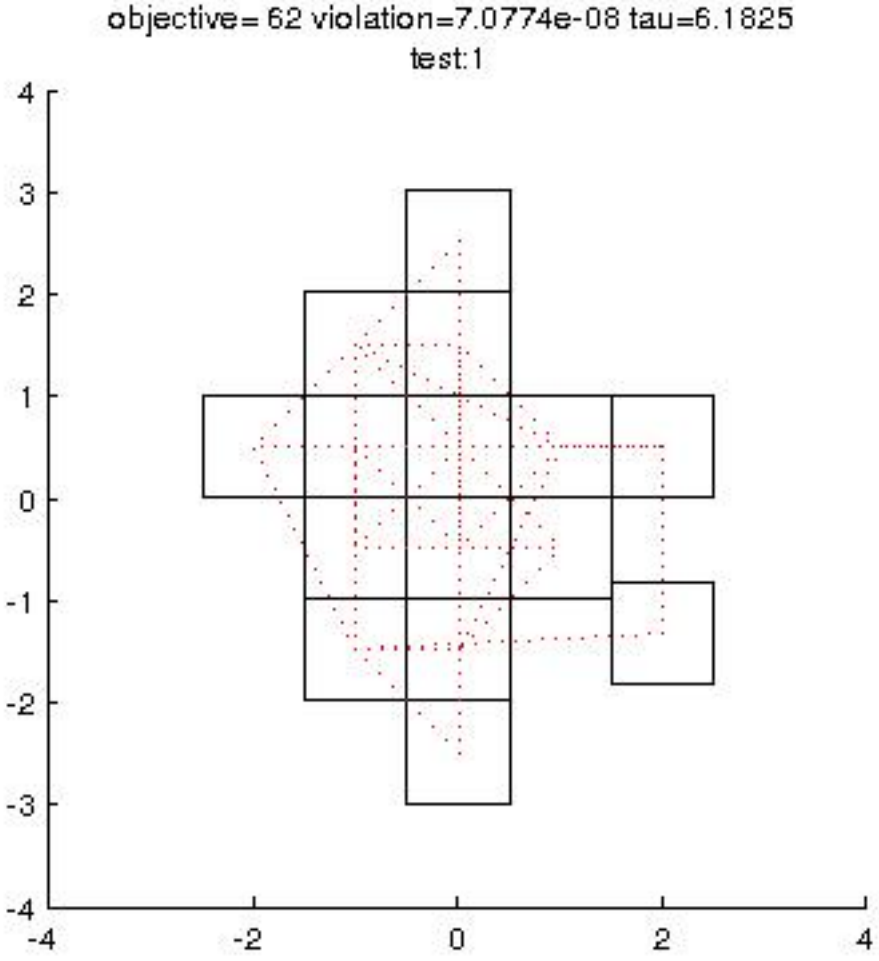
### 7.5.3 Problem Instance

For demonstration purposes, following instance is used for experimentation purpose :

$$
\begin{gathered}
n = 15 \text{ components}, \\
\text{Average degree of freedom} = 6, \\
l_i = 1, \\
b_x = b_y = 7, \\
\tau_0 = 0.2, \\
\tau_{max} = 10000, \\
\mu = 1.1
\end{gathered}
$$

### 7.5.4 Computational details

The sub-problems are solved using CVX on the system on a 3.2 GHz intel i3 processor. Flop required are mentioned and shown in section-7.7.



### 7.5.5 Results

For the specified problem instance, following was the outcome of the circuit placement problem. This result shows the final state of the system. More details in Section-7.7

## 7.6 Other Non-CCP methods for Placement Problem

This section compares the method proposed in [2] with other non-ccp methods for solving circuit placement problem.

1. [4] demonstrates an approach for Circuit placement problem using Interior point method for NP and the fast Multipole method. This approach to tackle the problem ensures the termination in (n*(n-1))/2 steps for n-components. In CCP-method, this is not the case, as it is clearly visible from section-7. This the method proposed in [4] is more scalable and can be used for solving large systems.

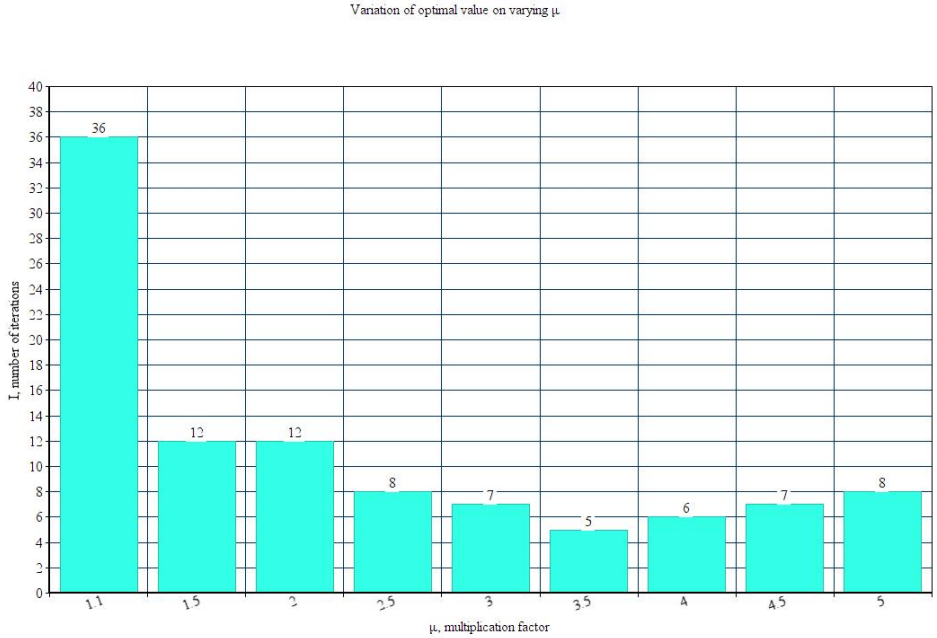
2. [5] presented an approach for FPGA circuit placement problem using min-cut bipartitioning based method. The method presented in [5] also takes into account the routability of circuit along with non-overlapping constraint. Thus for the case of FPGA circuits, the method proposed in [5] is a better approach than CCP-method demonstrated in this term paper.

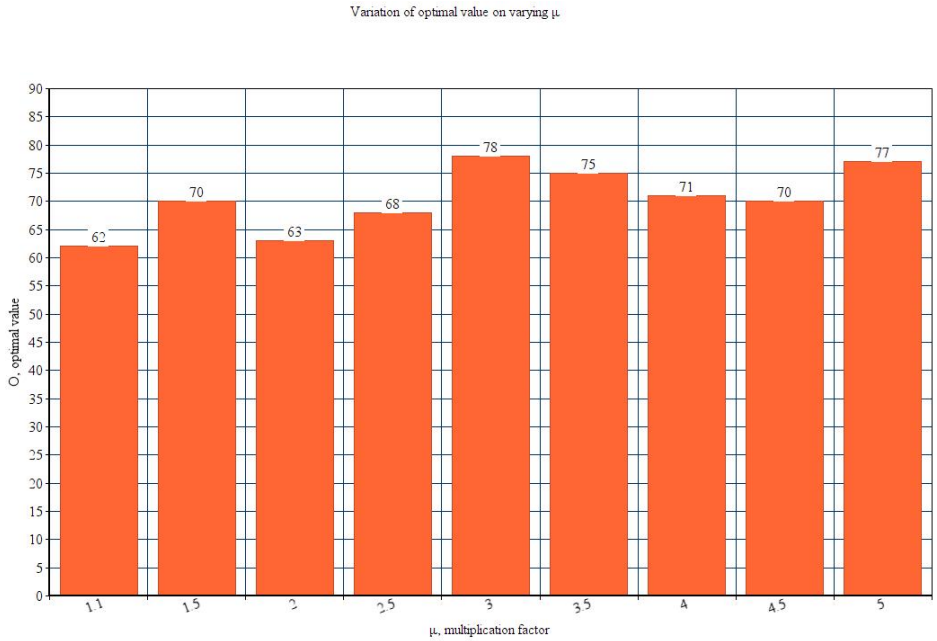
## 7.7 Experimentation and Observations

Following section involves analysis of optimal value and effect on number of iterations on varying various parameters of the algorithm.
$\mu$ = factor with which penalty is multiplied in each iteration.
$\tau$ = Initial penalty factor.
**Note -**

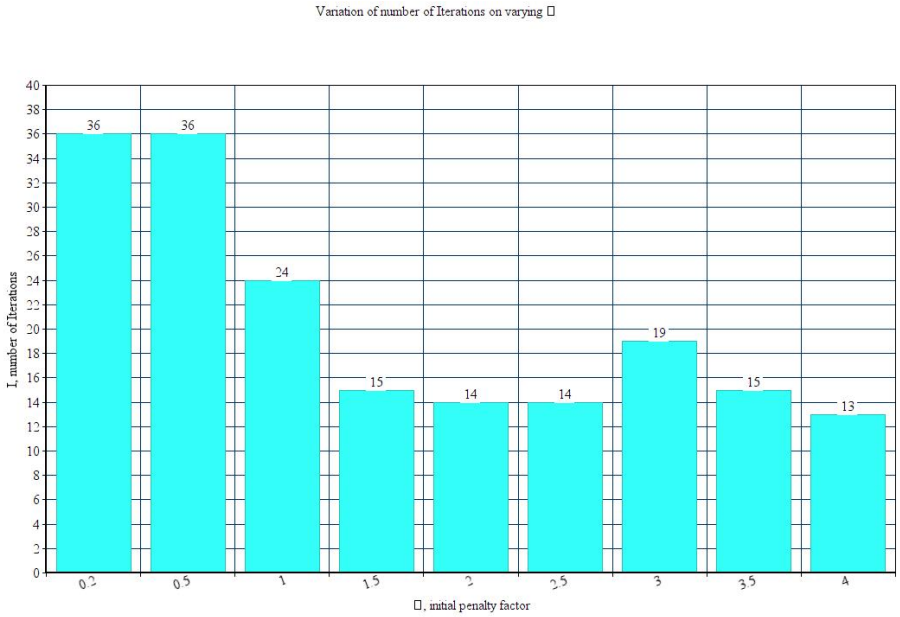
- For more on $\tau$ and $\mu$ refer to section-7.4.
- All the experiments are carried for the problem instance specified in section-7.5.3.
- For each experiment, only one parameter is varied at a time, rest all are kept are fixed at values mentioned in section-5.3.

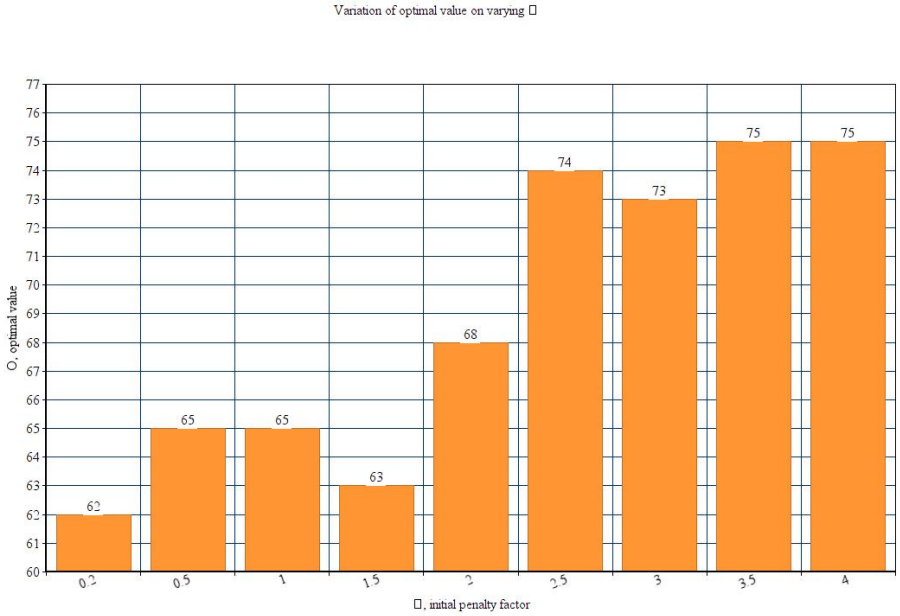
### 7.7.1 Effect on optimal value and number of iterations on varying $\mu$



### 7.7.2 Effect on optimal value and number of iterations on varying $\tau$




1

[1]Yash Patel -201301134
[1]Noor Pratap Singh-201364176

# 8 Conclusion

# References


[1] K. Grove-Rasmussen og Jesper Nygård, *Kvantefænomener i Nanosystemer.* Niels Bohr Institute & Nano-Science Center, Københavns Universitet

[2] Caldwell,Kahng,Markov *Circuit Placement* 1991.

[3] Thomas Lipp, Stephen Boyd *Variations and Extension of the Convex-Concave Procedure*

[4] Mikhail Belkin, Partha Niyogi *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation:* Department of Mathematics & Department of Computer Science and Statistics, University of Chicago

[5] Tony.F.Chan, Jason Cong *Multilevel optimization for Large-Scale Circuit placement*

[6] Zoltan Baruch, Octavian Creţ, Kalman Pusztai *AN EFFICIENT SEQUENCE TO APPLY SLICING LINES IN FPGA PLACEMENT:*