DIGITAL CIRCUIT

A Mathematical

Programing Report

Submitted in the partial fulfillment of the

requirements for the award of the degree of

Bachelor of Technology in

Department of Computer Science and Engineering

By

Krishna sai Roshith (2010030283)

Harshith naidu(2010030515)

P.hitesh(2010030510)

Priyanshu (2010030324)

under the supervision of

Dr. Anal Paul

Assistant professor

Department of Computer Science and Engineering

K L University Hyderabad,

Aziz Nagar, Moinabad Road, Hyderabad – 500 075, Telangana, India.

March, 2022

Declaration

The Mathematical Programming-II report entitled "**DIGITAL CIRCUIT** " is a record of bonified work of Roshith (2010030283), Harshith naidu(2010030515) ,P.Hitesh(2010030510),Priyanshu (2010030324) submitted in partial fulfillment for the award of B.Tech in the Department of Computer Science and Engineering to the K L University, Hyderabad. The results embodied in this report have not been copied from any other Departments/ University/ Institute.

Certificate

This is to certify that the Mathematical Programming-II Report entitled "**DIGITAL CIRCUIT**" is being submitted Roshith (2010030283),Harshith naidu (2010030515) ,P.Hitesh(2010030510), Priyanshu (2010030324) submitted in partial fulfillment for the award of B.Tech in CSE to the K L University, Hyderabad is a record of bonafide work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other departments/ University/Institute.

Signature of the Supervisor

Dr. Anal paul

Assistant Professor

Signature of the HOD

Signature of the External Examiner

ACKNOWLEDGEMENT

First and foremost, we thank the lord almighty for all his grace & mercy showered upon us, for completing this Mathematical Programming-II report successfully.

We take grateful opportunity to thank our beloved Founder and Chairman who has given constant encouragement during our course and motivated us to do this Social Internship. We are grateful to our Principal **Dr. L. Koteswara Rao** who has been constantly bearing the torch for all the curricular activities undertaken by us.

We pay our grateful acknowledgement & sincere thanks to our Head of the Department **Dr. Chiranjeevi Manike** for his exemplary guidance, monitoring, and constant encouragement throughout the course of the Mathematical Programming-II. We thank Mr. Anal Paul of our department who has supported throughout MP-II holding a position of supervisor.

We whole heartedly thank all the teaching and non-teaching staff of our department without whom we won’t have made this MP-II a reality. We would like to extend our sincere thanks especially to our parent, our family members and friends who have supported us to make this a grand success.

ABSTRACT

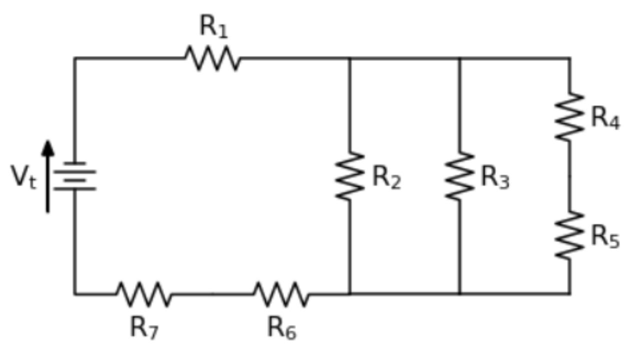
A method for digital circuit optimization based on formulating the problem as a geometric program (GP) or generalized geometric program (GGP), which can be transformed to a convex optimization problem and then very efficiently solved. We start with a basic gate scaling problem, with delay modeled as a simple resistor-capacitor (RC) time constant, and then add various layers of complexity and modeling accuracy, such as accounting for differing signal fall and rise times, and the effects of signal transition times. We then consider more complex formulations such as robust design over corners, multimode design, statistical design, and problems in which threshold and power supply voltage are also variables to be chosen. Finally, we look at the detailed design of gates and interconnect wires, again using a formulation that is compatible with GP or GGP. Subject classifications: programming: geometric; engineering: computer-aided design; digital circuit optimization

Introduction

The complexity of digital integrated circuits (ICs) has been increasing exponentially since around 1960, with the number of components or devices in a single IC more than doubling every 18 months. Some current ICs contain over 100 million devices and a similar number of wires connecting them. The design of such complex ICs relies heavily on electronic design automation (EDA) and computer-aided design (CAD) technologies. we focus on just one step in the design of a digital circuit: the selection of appropriate sizes for the devices, gates, and wires. These sizes can correspond to physical dimensions such as the width of a transistor channel or wire segment, or to more abstract parameters like the *drive strength* of a gate (which is closely related to the physical size of the gate). We will also consider extensions in which other design variables, such as threshold and power supply voltage, are also selected. The choice of these design variables can have a strong influence on the three primary top level objectives: the total area of the circuit, the total power it consume s, and the speed at which it can operate.

Methodology and implementation

The circuit diagram below with a driving voltage $V_t = 5.20V$ and resistor values in the table below.



A table of resistance values is below:

V_t =	5.20 V
R_1 =	13.2 mΩ
R_2 =	21.0 mΩ
R_3 =	3.60 mΩ
R_4 =	15.2 mΩ
R_5 =	11.9 mΩ
R_6 =	2.20 mΩ
R_7 =	7.40 mΩ

Solutions :-

In [17]:
```
pip install pyswarms
```

```
Collecting pyswarms
  Using cached pyswarms-1.3.0-py2.py3-none-any.whl (104 kB)
Requirement already satisfied: future in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (0.18.2)
Requirement already satisfied: numpy in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (1.20.3)
Requirement already satisfied: pyyaml in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (6.0)
Requirement already satisfied: attrs in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (21.2.0)
Requirement already satisfied: scipy in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (1.7.1)
Requirement already satisfied: tqdm in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (4.62.3)
Requirement already satisfied: matplotlib>=1.3.1 in c:\users\dell\anaconda3\lib\site-packages (from pyswarms) (3.4.3)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\dell\anaconda3\lib\site-packages (from matplotlib>=1.3.1->pyswa
rms) (2.8.2)
Requirement already satisfied: pillow>=6.2.0 in c:\users\dell\anaconda3\lib\site-packages (from matplotlib>=1.3.1->pyswarms)
(8.4.0)
Requirement already satisfied: cycler>=0.10 in c:\users\dell\anaconda3\lib\site-packages (from matplotlib>=1.3.1->pyswarms) (0.
10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dell\anaconda3\lib\site-packages (from matplotlib>=1.3.1->pyswarm
s) (1.3.1)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\dell\anaconda3\lib\site-packages (from matplotlib>=1.3.1->pyswarms)
(3.0.4)
Requirement already satisfied: six in c:\users\dell\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib>=1.3.1->pyswarm
s) (1.16.0)
Requirement already satisfied: colorama in c:\users\dell\anaconda3\lib\site-packages (from tqdm->pyswarms) (0.4.4)
Installing collected packages: pyswarms
Successfully installed pyswarms-1.3.0
Note: you may need to restart the kernel to use updated packages.
```

In [18]:
```python
import sys
import numpy as np
import matplotlib.pyplot as plt

# Import PySwarms
import pyswarms as ps
```

In [19]:
```python
print('Running on Python version: {}'.format(sys.version))
```

```
Running on Python version: 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
```

In [20]:
```python
def cost_function(I):

    #Fixed parameters
    U = 10
    R = 100
    I_s = 9.4e-12
    v_t = 25.85e-3

    c = abs(U - v_t * np.log(abs(I[:, 0] / I_s)) - R * I[:, 0])

    return c
```

In [21]:
```python
%%time
# Set-up hyperparameters
options = {'c1': 0.5, 'c2': 0.3, 'w':0.3}

# Call instance of PSO
optimizer = ps.single.GlobalBestPSO(n_particles=10, dimensions=1, options=options)

# Perform optimization
cost, pos = optimizer.optimize(cost_function, iters=30)
```

```
2022-05-06 11:33:16,303 - pyswarms.single.global_best - INFO - Optimize for 30 iters with {'c1': 0.5, 'c2': 0.3, 'w': 0.3}
pyswarms.single.global_best: 100%|██████████████████████████████████████████████████████████|30/30, best_cost=0.000168
2022-05-06 11:33:16,501 - pyswarms.single.global_best - INFO - Optimization finished | best cost: 0.0001683673802208574, best p
os: [0.09404937]
```

```
Wall time: 232 ms
```

In [22]:
```python
print(pos[0])
```

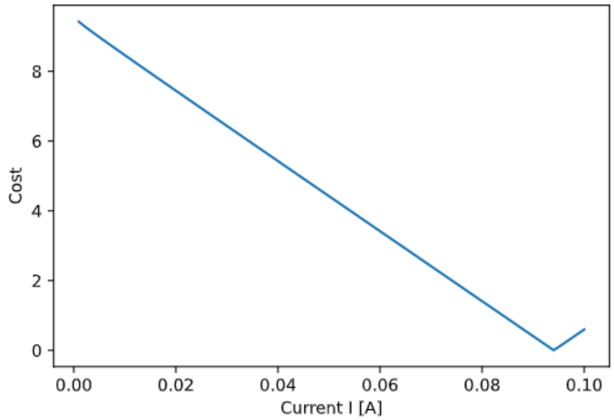
```
0.0940493654889524
```

In [23]:
```python
print(cost)
```

```
0.0001683673802208574
```

In [24]:
```
x = np.linspace(0.001, 0.1, 100).reshape(100, 1)
y = cost_function(x)

plt.plot(x, y)
plt.xlabel('Current I [A]')
plt.ylabel('Cost');
```



In [25]:
```
# Import non-linear solver
from scipy.optimize import fsolve
```

In [26]:
```
c = lambda I: abs(10 - 25.85e-3 * np.log(abs(I / 9.4e-12)) - 100 * I)

initial_guess = 0.09

current_I = fsolve(func=c, x0=initial_guess)

print(current_I[0])
```

```
0.09404768643017938
```

In [ ]:

APPLICATIONS

1.we can solve even large GPs very effectively, using recently developed methods.

2. so once we have a GP formulation, we can solve circuit design problem effectively.
The reason we of using it.

3. GP is especially good at handling a large number of concurrent constraints. 2. GP formulation is useful even when it is approximate

SOFTWARE REQUIRED

- PyCharm Community Version
- Python 3.10
- Jupyter Notebook

APPROACH

- Most problems don't come naturally in GP form; be prepared to reformulate and/or approximate.
- GP modeling is not a "try my software" method; it requires thinking.

Our Approach:

- Start with simple analytical models (RC, square-law, Pelgrom,...) to verify GP might apply.
- Then fit GP-compatible models to simulation or measured data.

For highest accuracy, revert to local method for final polishing.

Conclusion

- Comes up in a variety of circuit sizing contexts.
  - Can be used to formulate a variety of problems.
  - Admits fast, reliable solution of large-scale problems
  - Is good at concurrently balancing lots of coupled constraints and objectives.
  - Is useful even when problem has discrete constraints.