Natasa Miskov-Zivanov
Electrical & Computer Engineering
nmiskov@andrew.cmu.edu

Carnegie Mellon

760 Project 2: BUILD A PLACER

- **What you get from us**
  - Placer benchmarks, a range of real industrial netlists, sizes 20-12K gates
  - Info about the cell sizes, chip image size, pin locations around edges, etc
  - A conjugate gradient function minimizer (probably in C++ and in MATLAB)
  - A simple "Checker" program which will compute stats for each of your placer outputs: what is the actual wirelen, a simple version of the density penalty

- **What you do for us**
  - Read in the netlist
  - Set up and run your own version of an "analytical placer"
  - Place the netlist, which results in a set of **(xc, yc)** coords for all cells
  - Dump a file in a simple format with all this **(xc, yc)** info

Project 2 Details: Cost Function and Gradient

- **Note: you have to compute cost f(…xi…, …yi…) (a scalar)**
- **…AND have to compute gradient f' (…xi…, …yi…) (a vector)**
  - Unlike the little example we showed, it's not a closed-form formula
  - You have to compute each element of the gradient vector *numerically:*

$$\partial f / \partial xi = \frac{f(xi + h) - f(xi)}{h} \qquad \partial f / \partial yi = \frac{f(yi + h) - f(yi)}{h}$$

  - You just move **cell(i)** a very small amount on **x** direction to get **f(xi+h)**
  - You just move **cell(i)** a very small amount in **y** direction to get **f(yi+h)**
  - **h** should be some *very small fraction* of the **gridlength**

- **Means you should think about to eval f( ) and f' ( ) *incrementally*, without recalculating every term every time**

Project 2 Details: Netlist Format

- **Chip geometry**
  - Image is always **100x100**, with **(0,0)** at lower left
  - Pins around the edge of the chip with specified coordinates on boundary

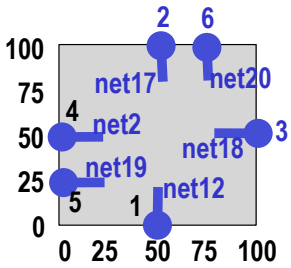
- **Cells**
  - Each cell is **1** unit high and **(#nets it connects to)** units wide
  - To compute area of cell: **area = (fundamental_unit) * (#nets it connects to)**
  - We give you the fundamental_unit number at start of the file

- **Utilization**
  - Always **75%** for all these benchmarks (this is pretty easy for a real ASIC)
  - Means 75% of 100x100 chip area should be filled with gates when done

Project 2 Details: Netlist Input Format

```
100 100 147.05882329  // ChipXWidth ChipYHeight FundamentalUnit
18 20                 // NumberofGates NumberofNets
1 2 20 19             // Gate# NumNetsConnected NetNumber...
2 2 2 1               // Example: Gate#2 is connected to two nets: #2 & #1
3 2 1 18
4 3 3 4 5
5 6 6 5 4 3 7 8
6 2 9 4
7 2 10 3
8 2 19 9
9 2 9 10
10 4 11 10 12 13
11 2 12 16
12 2 7 6
13 2 12 11
14 2 14 17
15 2 8 15
16 5 15 16 17 18 7
17 4 16 17 18 14
18 5 13 16 17 18 12
6                     // Number of Pins connected to nets
1 12 25 0             // PinID NetNumberConnectedTo PinX PinY
2 17 50 100           // Example: Pin #2 is connected to Net #17 at
                      // coordinates (2,4)
3 18 100 50
4 2 0 50              // NOTE: The above comments will NOT
5 19 0 25             // be in the REAL input files.
6 20 75 100
```

Cell 10 is 1 unit high x 4 units wide
Area = 1 x 4 x 147.05882329 = 588.235



Project 2 Details: Output File Format

- Simple. If you have n gates, you write out n lines, each of which looks like

```
integerGATEID  floatXcoord floatYcoord \n
```

- That's it. n gates, n lines, gateID and float (xc, yc) per line

- Aside:
  - Yes, as far as we know, the gate ID nums start at 1 and go up consecutively
  - The net ID nums should do the same (there might be some holes…)

18-760 Project 2: Analytical Placer

- **As before, it's a *bad* idea to just let you go for ~4 weeks and hope it works. Intermediate deadline *forces* an earlier start.**

30	M	ASIC Placement 1: cont.	
April			
1	W	ASIC Placement 2: Recursive	**Proj2 Out**
6	M	ASIC Placement 3: Analytical	
8	W	ASIC Routing; Project 2 Review	
13	M	Timing Analysis: Static Timing	
15	W	Timing Analysis: Electrical Timing, Elmore delay	**Intermediate milestone**, **Wed Apr 22** midnight Show *some progress* on TOY benchmarks
20	M	Elmore delay, Geometric data structures	
22	W	Geometric Data Structures	
27	M	TBD	
29	W	EXAM II, 2 hours in class, open notes	
May			
1	F	Last Day of Classes	**Final deadline**, **Wed May 6** midnight. Able to run real benchmarks
4-11		Final Exam Period	

18-760 Proj 2 Placer: Grading [100 pts]

- **[20 points] Progress @ Intermediate Milestone**
  - We want to see what happens when you try to run the TOY benchmarks
  - These are tiny < 100 gate designs that should pose no "complexity" issues
  - The example on the previous slide is what you should start with
  - We don't need to see a great layout, we just want to see "progress"
  - Advice: get your cost function and gradient calculations done asap, even if they are flat (not incremental), put this in the optimizer, and start playing with the magic parameters to see how it works. Don't try to do an "outer" loop yet – just make the optimizer run once, and see what you get
- **[20 points] Final Placer Performance**
  - We are going to run your code on a series of benchmarks. Some will be tiny, some medium; we'll see what happens.
  - You will be graded on what your placer can do, and how well: wirelength, and how well the gates spread out.

18-760 Proj 2 Placer: Grading [100 pts]

- **[20 points] *Coolness***
  - These points will be awarded for the "impressiveness" of your project. Since this project is less about "correctness" than the SAT project, there's lots of options for scoring "coolness"
  - **Examples:** Sophisticated inner/outer loops. Can run big benchmarks. Better wirelength and density. Some sort of graphics.

- **Aside: about "*coolness*"**
  - You can try to aim for some fairly **simple**, but do it really well, fully working, with lots of great experimental verification. Since **lots** of degrees of freedom in this placer, you can get points for showing how you explored these.
  - You can try to aim for something very **aggressive**, and maybe not get all of it working perfectly, but enough (*with results*) to convince us you understand it

Carnegie Mellon

18-760 Proj 2 Placer: Grading [100 pts]

- **[40 points] Write-up**
  - This is the most important, and thus most heavily-weighted, part of the project. Your report should be spell-checked, typed, grammatically correct, and readable, sent in as pdf. You must describe, in lucid detail, what you planned to build, what you built, and how well it worked:
    - **[8 pts] Goals:** The goals you set out to achieve
    - **[8 pts] Placer Features:** Each idea you implemented – your description of each idea must be sufficient to convince us that you fully understand how that idea is supposed to work, and that you really (tried to) implement it reasonably. **(**Include anything you feel is relevant or noteworthy about your project that will help us understand what you accomplished.)
    - **[8 pts] Results:** Your experimental results, *especially* tests that you chose to explain what *benefit* you got from your "cool" features.
    - **[16 pts]** Quality of writeup: Professional, correct, polished, etc.

Options for: Graphical Output

- **Hey – it's a placer, it's geometric, it's interesting to watch run**
  - What graphics you do - up to you
  - How you do it is also up to you
  - We will supply a tcl/tk app that let's you just print "drawing" instructions to a file in a very simple format, so you can make pics, movies, animations, etc.
  - We will also supply this "viewer" application that let's you interactively browse and control the graphical outputs
  - You just need to get the tcl/tk distribution for your machine, and be able to run a wish shell on your machine. Should run on Linux, Mac, Win