Optimizing Technology Mapping by Limiting Standard Cell Selection

Daniel Robinson - ECE 5740

Introduction

- Standard cell libraries can have thousands of different cells
- Most designs don’t need many of these cells for an efficient mapping
- Removing some cells from the library can result in mappings with less delay and area when using heuristic mapping methods like ABC

Setup

- Used ABC for testing
- Library with 161 cells
- 22k line ode.abc.blif design file used for testing

```
# The genlib library "ASAP7_7nm_LVT_FF.genlib" with 161 gates written by ABC on Mon Apr 10 12:43:35 2023
GATE _const0_                0.00  z=CONST0;
GATE _const1_                0.00  z=CONST1;
GATE BUFx10_ASAP7_75t_L      3.27  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx12_ASAP7_75t_L      3.73  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx12f_ASAP7_75t_L     4.20  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx16f_ASAP7_75t_L     5.13  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx24_ASAP7_75t_L      7.00  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx2_ASAP7_75t_L       1.17  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx3_ASAP7_75t_L       1.40  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx4_ASAP7_75t_L       1.63  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx4f_ASAP7_75t_L      1.87  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx5_ASAP7_75t_L       1.87  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx6f_ASAP7_75t_L      2.33  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE BUFx8_ASAP7_75t_L       2.80  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HB1xp67_ASAP7_75t_L     0.93  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HB2xp67_ASAP7_75t_L     1.17  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HB3xp67_ASAP7_75t_L     1.40  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HB4xp67_ASAP7_75t_L     1.63  Y=A;                                                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx11_ASAP7_75t_L      3.03  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx13_ASAP7_75t_L      3.50  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx1_ASAP7_75t_L       0.70  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx2_ASAP7_75t_L       0.93  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx3_ASAP7_75t_L       1.17  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx4_ASAP7_75t_L       1.40  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx5_ASAP7_75t_L       1.63  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx6_ASAP7_75t_L       1.87  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVx8_ASAP7_75t_L       2.33  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVxp33_ASAP7_75t_L     0.70  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE INVxp67_ASAP7_75t_L     0.70  Y=!A;                                                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND2x2_ASAP7_75t_L      1.40  Y=(A * B);                                                            PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND2x4_ASAP7_75t_L      2.33  Y=(A * B);                                                            PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND2x6_ASAP7_75t_L      2.80  Y=(A * B);                                                            PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND3x1_ASAP7_75t_L      1.40  Y=(A * B * C);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND3x2_ASAP7_75t_L      1.63  Y=(A * B * C);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND3x4_ASAP7_75t_L      3.73  Y=(A * B * C);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND4x1_ASAP7_75t_L      1.63  Y=(A * B * C * D);                                                    PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND4x2_ASAP7_75t_L      3.73  Y=(A * B * C * D);                                                    PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND5x1_ASAP7_75t_L      1.87  Y=(A * B * C * D * E);                                                PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE AND5x2_ASAP7_75t_L      4.67  Y=(A * B * C * D * E);                                                PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE FAx1_ASAP7_75t_L        3.27  CON=(!A * !B) + (!A * !CI) + (!B * !CI);                               PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE FAx1_ASAP7_75t_L        3.27  SN=(A * B * !CI) + (A * !B * CI) + (!A * B * CI) + (!A * !B * !CI); PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HAxp5_ASAP7_75t_L       2.10  CON=(!A) + (!B);                                                      PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE HAxp5_ASAP7_75t_L       2.10  SN=(A * B) + (!A * !B);                                               PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE MAJIxp5_ASAP7_75t_L     1.63  Y=(!A * !B) + (!A * !C) + (!B * !C);                                  PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE MAJx2_ASAP7_75t_L       2.10  Y=(A * B) + (A * C) + (B * C);                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE MAJx3_ASAP7_75t_L       2.33  Y=(A * B) + (A * C) + (B * C);                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE NAND2x1_ASAP7_75t_L     1.40  Y=(!A) + (!B);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE NAND2x1p5_ASAP7_75t_L   1.87  Y=(!A) + (!B);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
GATE NAND2x2_ASAP7_75t_L     2.33  Y=(!A) + (!B);                                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
```

Baseline

read 7nm.genlib; read ode.abc.blif; map; write temp.blif; read 7nm_lvt_ff.lib; read -m temp.blif; ps; topo; upsize; dnsize; stime;

Area = 9341.46

Delay = 906.43 ps

```
WireLoad = "none"  Gates =   9573 ( 20.5 %)   Cap =  1.1 ff (  2.9 %)
   Area =     9341.46 ( 87.4 %)   Delay =   906.43 ps  (  4.3 %)
```

Selection Method #1: Random Sampling

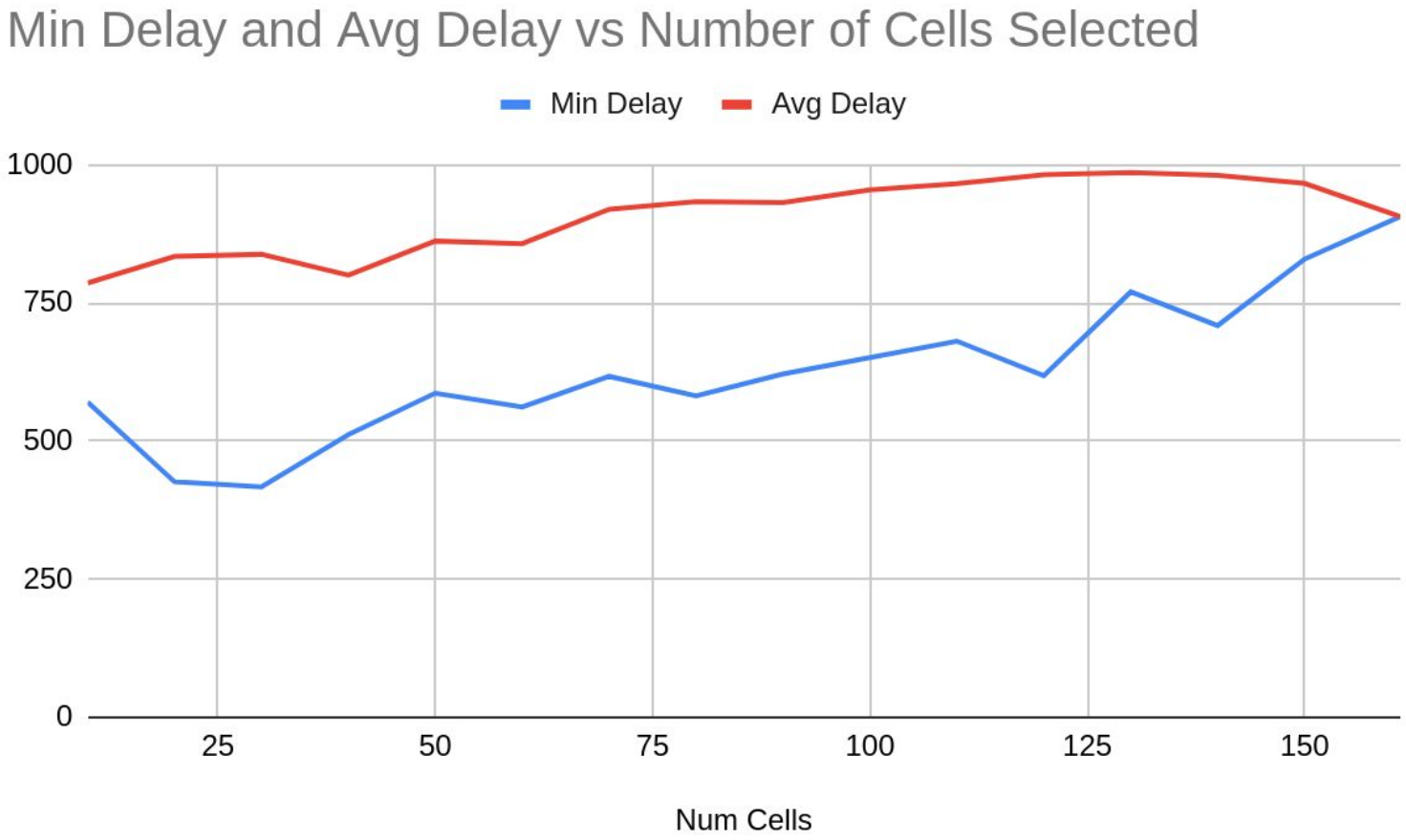
- We can randomly sample a certain number of cells from the library and construct a new library using only those cells.
- 20 cell example:

```
1 GATE OAI322xp33_ASAP7_75t_L        2.10  Y=(!A1 * !A2 * !A3) + (!B1 * !B2) + (!C1 * !C2);            PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
2 GATE OA333x1_ASAP7_75t_L           2.80  Y=(A1 * B1 * C1) + (A1 * B1 * C2) + (A1 * B1 * C3) + (A1 * B2 * C1) + (A1 * B2 * C2) + (A1 * B2 * C3) + (A1 * B3 * C1) + (A1 * B3 * C2) + (A1
  * B3 * C3) + (A2 * B1 * C1) + (A2 * B1 * C2) + (A2 * B1 * C3) + (A2 * B2 * C1) + (A2 * B2 * C2) + (A2 * B2 * C3) + (A2 * B3 * C1) + (A2 * B3 * C2) + (A2 * B3 * C3) + (A3 * B1 * C1) +
  (A3 * B1 * C2) + (A3 * B1 * C3) + (A3 * B2 * C1) + (A3 * B2 * C2) + (A3 * B2 * C3) + (A3 * B3 * C1) + (A3 * B3 * C2) + (A3 * B3 * C3); PIN *  UNKNOWN   1 999     1.00     0.00
  1.00     0.00
3 GATE BUFx2_ASAP7_75t_L             1.17  Y=A;                                                         PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
4 GATE OAI21xp5_ASAP7_75t_L          1.17  Y=(!A1 * !A2) + (!B);                                        PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
5 GATE AO22x1_ASAP7_75t_L            2.10  Y=(A1 * A2) + (B1 * B2);                                     PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
6 GATE BUFx3_ASAP7_75t_L             1.40  Y=A;                                                         PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
7 GATE XOR2x2_ASAP7_75t_L            2.57  Y=(A * !B) + (!A * B);                                       PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
8 GATE OA331x2_ASAP7_75t_L           2.57  Y=(A1 * B1 * C1) + (A1 * B2 * C1) + (A1 * B3 * C1) + (A2 * B1 * C1) + (A2 * B2 * C1) + (A2 * B3 * C1) + (A3 * B1 * C1) + (A3 * B2 * C1) + (A3
  * B3 * C1); PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
9 GATE HB3xp67_ASAP7_75t_L           1.40  Y=A;                                                         PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
10 GATE AO333x2_ASAP7_75t_L          3.03  Y=(A1 * A2 * A3) + (B1 * B2 * B3) + (C1 * C2 * C3);         PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
11 GATE OA222x2_ASAP7_75t_L          2.80  Y=(A1 * B1 * C1) + (A1 * B1 * C2) + (A1 * B2 * C1) + (A1 * B2 * C2) + (A2 * B1 * C1) + (A2 * B1 * C2) + (A2 * B2 * C1) + (A2 * B2 * C2); PIN
  *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
12 GATE NAND2xp5_ASAP7_75t_L         0.93  Y=(!A) + (!B);                                               PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
13 GATE OAI22xp5_ASAP7_75t_L         1.40  Y=(!A1 * !A2) + (!B1 * !B2);                                 PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
14 GATE OR4x2_ASAP7_75t_L            1.87  Y=(A) + (B) + (C) + (D);                                     PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
15 GATE OR5x2_ASAP7_75t_L            2.10  Y=(A) + (B) + (C) + (D) + (E);                               PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
16 GATE XNOR2xp5_ASAP7_75t_L         2.10  Y=(A * B) + (!A * !B);                                       PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
17 GATE BUFx16f_ASAP7_75t_L          5.13  Y=A;                                                         PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
18 GATE OR3x2_ASAP7_75t_L            1.63  Y=(A) + (B) + (C);                                           PIN *  UNKNOWN   1 999     1.00     0.00     1.00     0.00
19 GATE _const0_                     0.00  z=CONST0;
20 GATE _const1_                     0.00  z=CONST1;                                                  S
```

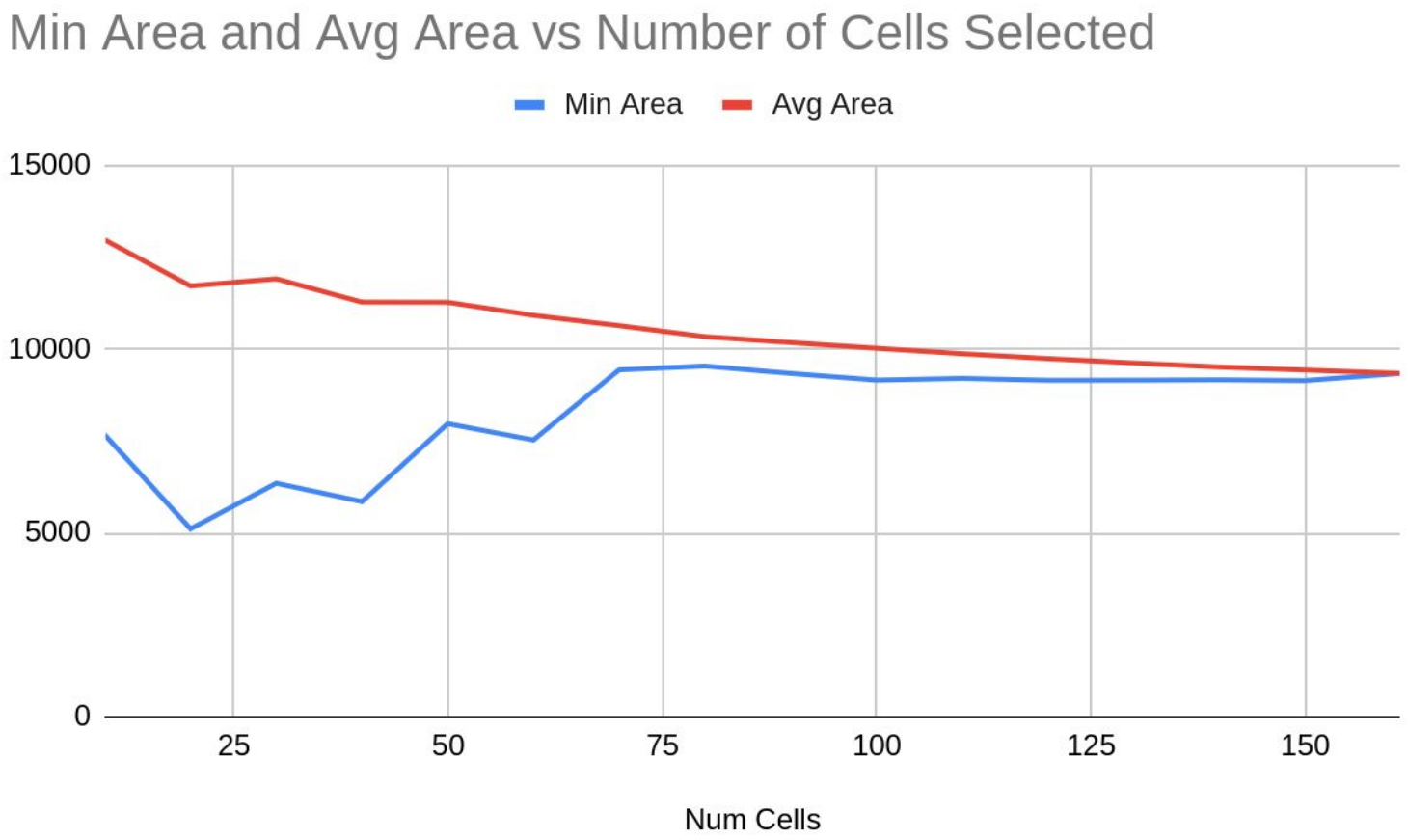
Random Sampling Test Methodology

- Create 1000 libraries with 10 randomly sampled cells, 1000 libraries with 20 randomly sampled cells, etc up to 150 randomly sampled cells
- Map the design using each library in ABC and record the minimum delays and areas, as well as the averages

Random Sampling Results



Random Sampling Results



Best Random Sampling Result

- I noticed that for this library, design, and mapping tool, selecting just ~20 cells resulted in the best results, so I tested 10k+ random samples of 20 cells
- Out of all those, the best selection resulted in a mapping with an area of just 1663 and a delay of just 135.73 ps.

```
WireLoad = "none"  Gates =   2315 ( 48.5 %)   Cap =  0.1 ff (  6.7 %)
   Area =     1663.05 ( 52.8 %)   Delay =   135.73 ps  (  1.0 %)
```

- Recall the baseline had an area of 9341.46 and a delay of 906.43 ps, so this demonstrates potential of 5x gains in some situations

```
WireLoad = "none"  Gates =   9573 ( 20.5 %)   Cap =  1.1 ff (  2.9 %)
   Area =     9341.46 ( 87.4 %)   Delay =   906.43 ps  (  4.3 %)
```

Selection Method #2: Sequential Optimization

1. Begin with the full library selected
2. Check all possible cells to remove, and remove the one that gives the best improvement to area or delay
3. Repeat until none of the cells give an improvement when removed

Using this method resulted in a delay of 685.07

Similar method:

1. Begin with full library selected
2. Loop through cells, try removing each one, and only keep the change if it results in better delay or area
3. Repeat step 2 starting at the first cell in the library each time until none of the removals give an improvement.

Using this method resulted in a delay of 700.78, but was much faster

Selection Method #3: DQN Reinforcement Learning

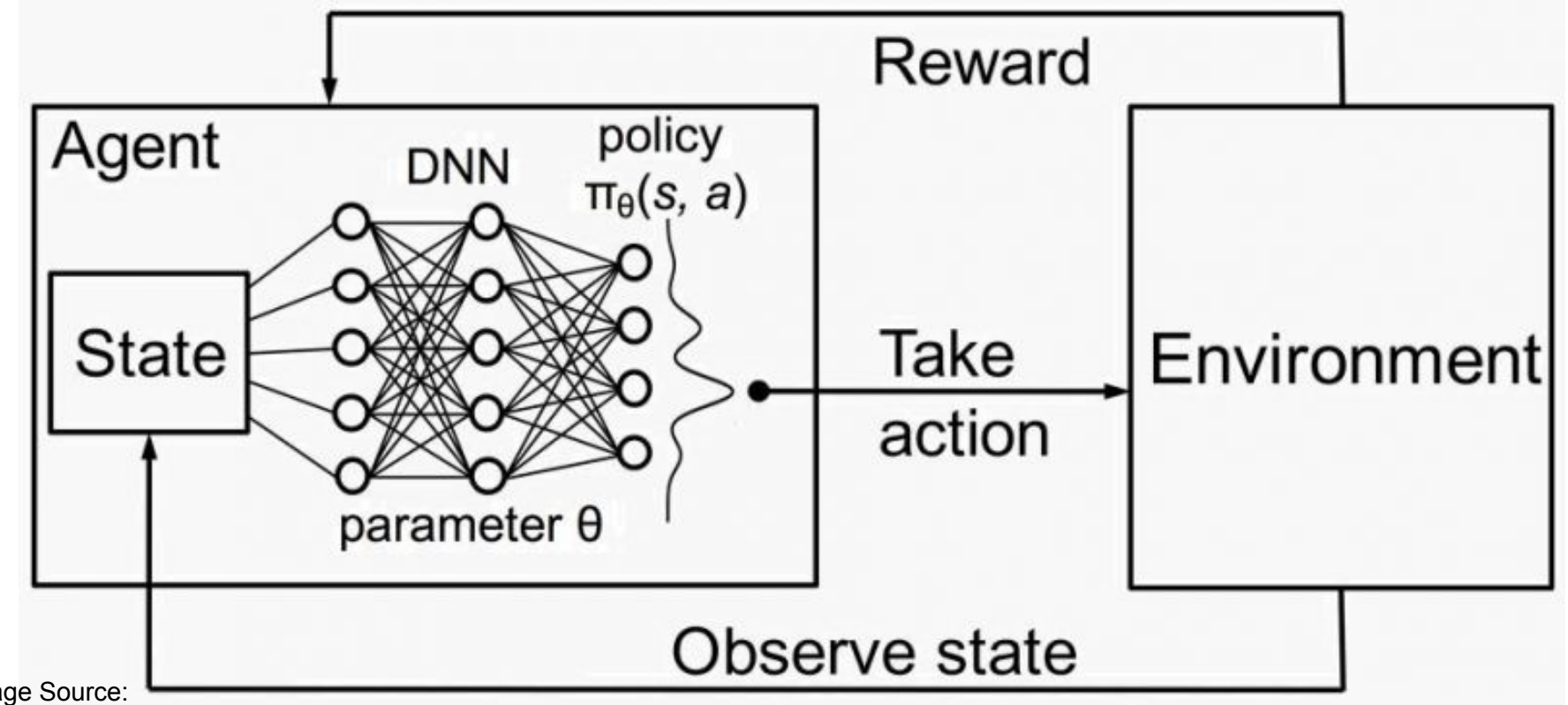


Image Source:
https://medium.com/@vishnuvijayanpv/deep-reinforcement-learning-value-functions-dqn-actor-critic-method-backpropagation-through-83a277d8c38d

Sequential DQN Setup

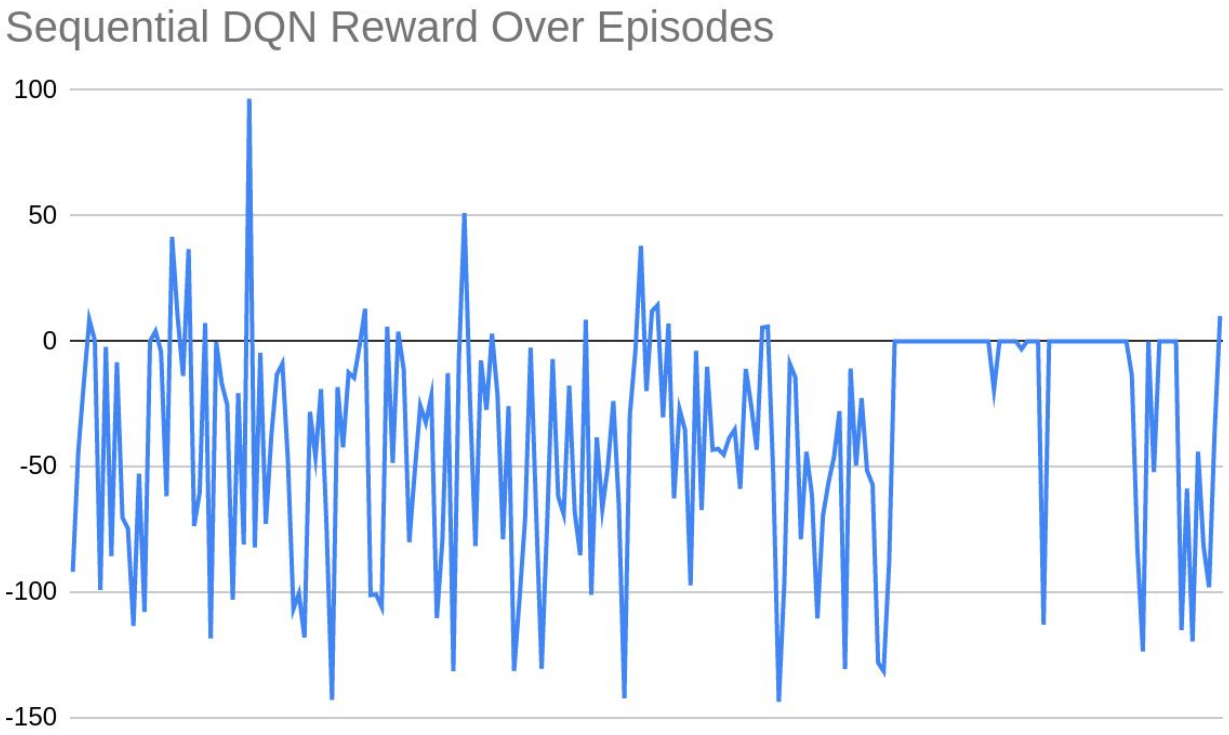
Environment:

- Setup with path to full library, path to design, and goal (min area or min delay)
- On setup and reset it has a state with all cells selected
- For each step it takes an input action which is a cell to change (add or remove), and it returns a reward based on the mapping from the new change

Agent:

- The neural network takes in the current environment state (selected cells) and outputs the action that it predicts will lead to the best reward
- Both the input and output layers have a number of nodes equal to the number of cells in the original library

Sequential DQN Results

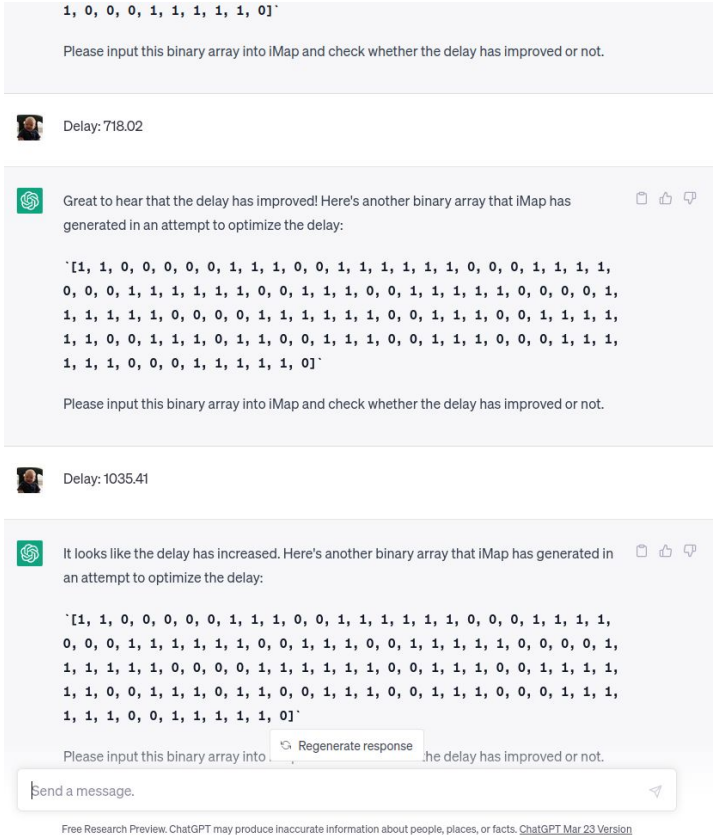


Alternate DQN Implementation

- Instead of the neural net selecting just one cell to change with its output, the action can be to select all the desired cells. Same neural net dimensions.
- I’ve been having trouble getting the loss function working properly for this idea, but in theory it is not much harder to implement than the sequential DQN.

Selection Method #4: ChatGPT, just for fun

- Told ChatGPT to minimize delay by passing in boolean arrays which represent its selection.
- It worked for a while, but for a longer time it would likely get off track or forget previous information



Conclusion

- There appears to be potential to better optimize technology mapping by strategically reducing the number of cells in the standard cell library
- With many attempts, random sampling works quite well to reduce delay and area significantly
- More strategic methods may lead to more efficient or more consistent results
- Further work is needed to better understand the potential of this idea