Michael Wang
CS 181 HW 1

0) The Sipser textbook is split into three parts each of which contains a few chapters. Each chapter is composed of several subsections, which are numbered one up. Throughout each chapter are figures, examples, and theorems, all of which share the same numbering scheme increasing from one up regardless of chapter subsection. For example, a theorem following example 27 is labled theorem 28, even if the example is at the end of one subsection and the theorem at the beginning of the next (except if the chapter ends). At the end of each chapter are a set of excercises followed by a set of problems. These start at one and the problems begin counting where the excercises leave off.

1a) Connecting any two graphs results in a single graph. The resulting graph must be connected as any point in G can reach any point in H by going to the node in G, crossing the new edge to the node in H, and going to the destination, with the reverse also being true going from H to G. The resulting graph must be acyclic as in order to be cyclic there must exist a pair of nodes which have multiple paths to reach one another. However, adding a single edge between two previously disconnected nodes can only increase their number of paths to one. The number of paths within G or H are unaffected and all paths crossing between G and H necessarily go through the newly added node limiting the number of paths to 1.

1b) The resulting graph must be connected for the same reasoning as in 1a. The resulting graph must contain a cycle. Label the newly added edges (G1, H1) and (G2, H2). Since the edges are distinct, either G1 is not G2 or H1 is not H2 or both. Assume G1 is not G2.

2a) {{0,x,0}, {0,x,1}, {0,y,0}, {0,y,1}, {0,z,0}, {0,z,1}, {1,x,0}, {1,x,1}, {1,y,0}, {1,y,1}, {1,z,0}, {1,z,1}}

2b) {{0,0,x}, {0,0,y}, {0,0,z}, {0,1,x}, {0,1,y}, {0,1,z}, {1,0,x}, {1,0,y}, {1,0,z}, {1,1,x}, {1,1,y}, {1,1,z}}

2c) {ε, {x}, {y}, {z}, {x,y}, {x,z}, {y,z}, {xyz}}

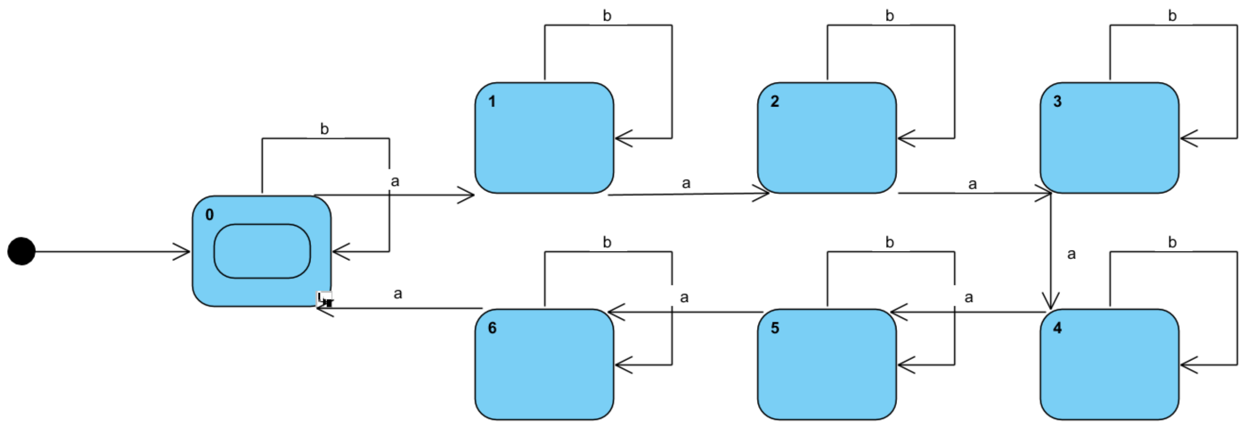
2d) {0x, 0y, 0z, 1x, 1y, 1z}

2e) {}

2f) {}

2g) {{x}, {y}, {z}, {1,x}, {1,y}, {1,z}}

3)



In this DFA, the nodes are labled by the number of observed {a} symbols modulo 7. Observing an {a} advances to the next node while observing a {b} simply loops back to the same node. The initial state is the accept state which is returned to after observing 7 {a}s.

4)
Q = {1odd, 1even, 0even, 00even, 0odd, 00odd, 000}
Σ = {0, 1}

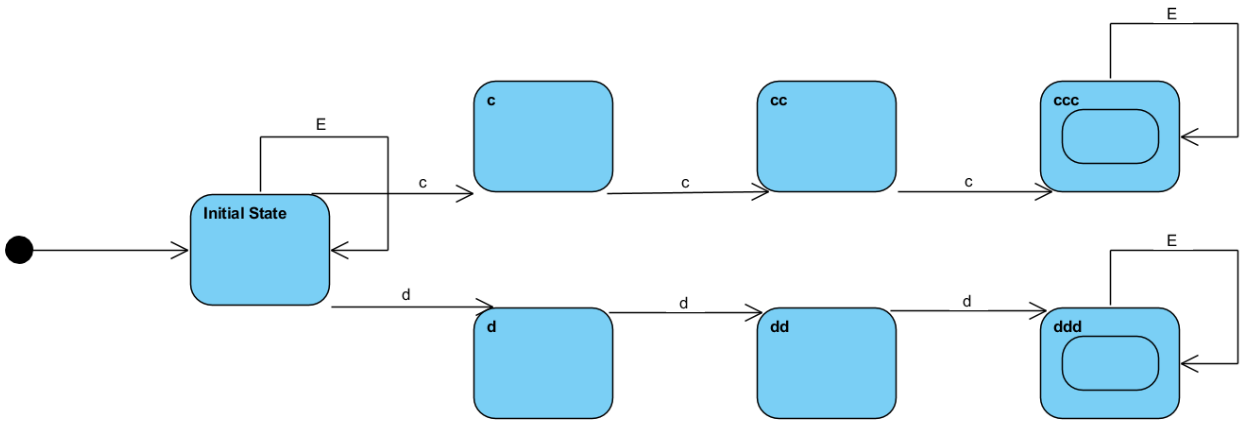
δ

	1even	1odd	0even	00even	0odd	00odd	000
0	0even	0odd	00even	000	00odd	000	000
1	1odd	1even	1even	1even	1odd	1odd	000

Start state is 1even
F = {1even, 0even, 00even}

In the DFA descrbed above every nodes but the last is labled by the most recently observed symbols along with the parity of the string so far. Since the relevant strings are 1, 0, and 00 and there are two parities, 6 states are formed this way. The final state is the rejected state which indicates a run of 3 zeros. In each state other than the reject state, encountering a 1 flips the parity and goes to the appropriate 1-state. Encountering a 0 proceeds to the next 0-state with the same parity.

5)



In this NFA, the nodes are labled by the most recent relevant symbols observed. An accept state is reached when exactly three {c}s or three {d}s are observed in a row, after which the accept state simply loops back on itself (E is used to represent Σ in the diagram).

6) The DFA recognizes words in which if the first symbol is 1 then the length of the word modulo 3 is 1, is the first symbol is 2 then the length of the word modulo 3 is 2, and if the first symbol is 0 then the length of the word modulo 3 is 0 (ie is divisible by 3).