Abstract
In this tutorial, we present optimal Cellular Nonlinear Network (CNN) templates for implementing linearly-separable one-dimensional (1-D) Cellular Automata (CA). From the gallery of CNN templates presented in this paper, one can calculate any of the 256 1-D CA Rules studied by Wolfram using a CNN Universal Machine chip that is several orders of magnitude faster than conventional programming on a digital computer.

1 Introduction
The main purpose of this paper is to derive the optimal Cellular Nonlinear Network (CNN) templates for linearly-separable one-dimensional (1-D) Cellular Automata (CA). A gallery of optimal CNN templates for linearly-separable 1-D CA is presented, and an appendix is included to illustrate the template derivation algorithm. These optimal templates may be implemented on any CNN universal chip [Chua & Roska, 2002], thereby allowing faster calculation of any of the 256 1-D CA rules [Wolfram, 2002] by several orders of magnitude. Such great enhancement in speed will enable researchers on CA to conduct extensive simulations over much longer periods (e.g., over many trillions of iterations) of Wolfram’s class 3 and 4 CA rules than currently feasible.

A. Truth Tables and Boolean Functions
Boolean Functions are described by truth tables. For the purpose of this paper, the inputs to the truth tables are binary 3-tuples, \((x_{i-1}, x_i, x_{i+1})\). That is, each of \(x_{i-1}, x_i, x_{i+1}\) can be either 1 or 0, leading to the generation of \(2^3 = 8\) input combinations, listed in order as \((0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)\). Each 3-tuple input is mapped to a binary output \(y\) that is also either 1 or 0. Thus, there are \(2^8 = 256\) possible truth tables generated from 3 binary inputs. Each of these 256 truth tables corresponds to a unique Boolean Function, which is named as the decimal equivalent of the binary number formed by
concatenating the output of each row in the truth table in reverse order. For example, when mapping (0,0,0) to 0, (0,0,1) to 1, (0,1,0) to 1, (0,1,1) to 1, (1,0,0) to 0, (1,0,1) to 1, (1,1,0) to 1, (1,1,1) to 0, the outputs may be written as the 8-tuple binary code (0,1,1,0,1,1,0). Concatenating these outputs in reverse order results in the binary number 01101110, whose decimal equivalent is 110. The corresponding Boolean Function is thus named as 110, which equals $0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$. Figure 1 presents that truth table and the derivation of the name.

<table>
<thead>
<tr>
<th>$x_{i-1}$</th>
<th>$x_i$</th>
<th>$x_{i+1}$</th>
<th>$y_i$</th>
<th>$2^0$</th>
<th>$2^1$</th>
<th>$2^2$</th>
<th>$2^3$</th>
<th>$2^4$</th>
<th>$2^5$</th>
<th>$2^6$</th>
<th>$2^7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Binary number: $0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0$

Decimal identification number:

$$0 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = \boxed{110}$$

Figure 1. The derivation of Boolean Function 110 and its truth table

B. Cellular Automata

A cellular automaton is a collection of cells that iterates on a set of rules, creating a new generation of cells with each iteration. A 1-D CA is a string of cells. It is assumed that the boundary condition is periodic, so that the string of cells is effectively a ring of cells as illustrated in Figure 2 for the case of a 1-D CA made of 10 cells.
The set of rules that govern the cells of the 1-D CA are summarized by 256 3-tuple truth tables. Each truth table is a Boolean Rule that dictates a distinct 1-D CA pattern from a given initial condition. For the purposes of studying 1-D CA, the initial condition would be a string of cells of an arbitrary length greater than or equal to 3 cells. Each cell would indicate a value of “0” or “1”. To find the output of each cell of a particular 1-D cellular automaton according to a fixed Boolean Rule, the cell in question is seen as $x_i$, its left neighbor as $x_{i-1}$, and its right neighbor as $x_{i+1}$. The corresponding output $y_i$ mapped from the resulting $(x_{i-1}, x_i, x_{i+1})$ in the truth table would be the output of the cell $x_i$. For example, let \{1,1,0,1\} be a sample initial condition bit string of length 4 for a 1-D 4-cell CA, given the truth table of Boolean Function 110, usually dubbed \textit{local Rule 110} in literature. Consider the third cell that contains “0.” The left and right neighbor cells are the second and fourth cells, respectively, both containing “1.” The output of the third cell would be the output $y_i$ mapped from the 3-tuple (1,0,1) from the truth table of local Rule 110. According to the truth table, when $x_{i-1} = 1$, $x_i = 0$, and $x_{i+1} = 1$, the output $y_i$ is 1, and hence the output of the third cell is “1.” The same process is repeated for each cell listed in the initial condition, resulting in the new \textit{evolution} \{0,1,1,1\}, and
completes one iteration of the 1-D CA. This new evolution, \{0,1,1,1\} would be the new initial condition for the next iteration. If all the cells containing “1” are colored red, and all the cells containing “0” are colored blue, then a pattern begins to emerge from the evolutions after several iterations. The first iteration for local Rule 110 is derived in Figure 3.

Figure 3. Starting with a sample initial condition, the next generation is found for Boolean Function 110.

In order to discover the distinct 1-D CA pattern generated by a local Rule and given initial condition, one would have to run the CA through several iterations. Some local Rules even have never-ending patterns\(^1\) formed from almost all possible initial conditions, such as Rules 30 (binary code 00011110) and 110 (binary code 01101110), as illustrated by Figure 4, when assuming an infinite number of cells.

\(^1\) Even for just 100 cells, there are \(2^{100}\) distinct bit-string patterns.
Figure 4(a). The pattern generated from rule 30 for a random initial condition. Notice how there is no repeated iteration.
Figure 4(b). The pattern generated from rule 110 for a random initial condition. Notice how there is no repeated iteration.
A Personal Computer (PC), which only has limited processing speed, certainly will not return these patterns instantaneously, especially if the initial condition bit string increases in length. Investigating higher dimensions of CA also poses future problems in pattern finding with a PC. A CNN universal chip is ideal for automation of n-cell 1-D CA not exceeding the array size of the chip.\(^2\) Thus, a CNN provides a dynamical system that completely predicts the CA evolution for any initial condition. The CNN chip has high processing speed (it can process under 1 nanosecond per iteration!), low power dissipation, and parallel processing. [Chua et al, 2002] All that is needed to program a CNN universal chip to automate the pattern-generation is templates that describe the Boolean Rules. These templates are found from corresponding Boolean Cubes.

C. Boolean Cubes

Since the 8 possible outputs in each truth table is either 0 or 1, there are \(2^8 = 256\) possible Boolean Functions/Rules. Each of these 256 Boolean Rules can be represented as a three-dimensional Boolean Cube for better visualization and analysis. For the reader’s convenience, Table 1 from [Chua et al, 2003] is reprinted herein as Table 1. Note that the Cubes that have numbers labeled in red are *linearly-separable*, while the Cubes that have numbers labeled in blue are *linearly non-separable*; these terms will be clarified in the following Section D. The lower box describes the coloring of the vertices of the Cubes, and how the vertices’ coordinates are derived from substituting “-1” for “0” for use in determining an analytical formula. Please refer to Figure 5 in deriving the analytical truth table and hence the coordinates of the vertices.

\(^2\) The current commercially available CNN chip has an array size of 176 x 144 [Anafocus, 2007].
Table 1. Encoding 256 local rules defining a binary 1D CA onto 256 corresponding "Boolean Cubes" [Chua et al, 2003].
Table 1 (continued)

<table>
<thead>
<tr>
<th>$u_{i+1}^t$</th>
<th>$u_{i}^t$</th>
<th>$u_{i+1}^t$</th>
<th>$u_{i}^t$</th>
<th>$u_{i+1}^t$</th>
<th>$u_{i}^t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Rule N

$u_{i+1}^t (u_{i+1}^t, u_{i}^t, u_{i+1}^t) = -1 \Rightarrow \beta_k = 0$

$u_{i+1}^t (u_{i+1}^t, u_{i}^t, u_{i+1}^t) = 1 \Rightarrow \beta_k = 1$
Table 1 (continued)

<table>
<thead>
<tr>
<th>$N$</th>
<th>$u_{i-1}$</th>
<th>$u_i$</th>
<th>$u_{i+1}$</th>
<th>$u_{i+2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>3</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>$y_i$</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>$y_i$</td>
</tr>
</tbody>
</table>

Note: $N$ is the decimal equivalent of the binary number.
Table 1 (continued)

<table>
<thead>
<tr>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
<th>$\beta_3$</th>
<th>$\beta_4$</th>
<th>$\beta_5$</th>
<th>$\beta_6$</th>
<th>$\beta_7$</th>
<th>$\beta_8$</th>
</tr>
</thead>
</table>

$N =$ decimal equivalent of binary number

<table>
<thead>
<tr>
<th>vertex</th>
<th>$u_{i-1}$</th>
<th>$u'_i$</th>
<th>$u_{i+1}$</th>
<th>$u'_{i+1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>3</td>
<td>-1</td>
<td>1</td>
<td>-1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\gamma_1$</td>
</tr>
</tbody>
</table>

Rule N

$\alpha_{j,k}^{\ast} (u_{i-1}, u'_i, u'_{i+1}) = 1 \rightarrow \beta_k = 1$

$\alpha_{j,k}^{\ast} (u_{i-1}, u'_i, u'_{i+1}) = -1 \rightarrow \beta_k = 0$
To recapitulate, in determining the Cube, if all the symbolic 0 values in the truth tables are replaced as -1 for numerical analytical purposes, then the new input values \( u_{i-1}, u_i, u_{i+1} \) would be the coordinates of the Cube’s vertices, with \( u = x - 1 \). The vertices would be colored according to the associated output \( y_i \) such that each Rule is represented as a unique Cube; vertices would be colored blue if the associated output is -1, and vertices would be colored red if the associated output is 1. The corresponding analytic truth table and Boolean Cube for Rule 110 is presented in Figure 5.

Figure 5. Converting symbolic truth table into analytic truth table to derive Boolean Cube. The 8-bit binary string is the binary number whose decimal equivalent is the name of the local Rule.
D. Complexity Index

These 256 Boolean Cubes may be categorized according to complexity. Each Cube is assigned a complexity index $\kappa$, which is the minimum number of planes needed to separate red vertices to one side of the plane, and blue vertices to the other. This complexity index is determined by inspection of the color of the 8 vertices in the Boolean Cube. Cubes that have $\kappa = 1$ require only 1 plane to separate the red vertices from the blue, and are linearly separable, e.g. the Cube for Rule 16. Cubes that have $\kappa = 2$ require 2 planes, such as the Cube for Rule 18, and Cubes that have $\kappa = 3$ require 3 planes, such as the Cube for Rule 150, as shown in Figure 6. These Cubes that require more than 1 plane to separate the red and blue vertices are linearly non-separable. The highest complexity index possible for the 256 1-D CA Boolean Cubes is 3.

Figure 6. Rule 16 has $\kappa=1$, Rule 18 has $\kappa=2$, Rule 150 has $\kappa=3$. The 8-bit binary string above each Boolean Cube denotes the binary number whose decimal equivalent is the name of each local Rule.

There are 104 Boolean Cubes that are linearly-separable; the corresponding 104 Boolean Rules are listed in Table 2. Tables 3 and 4 list $\kappa=2$ and $\kappa=3$ Rules respectively. To program a CNN universal chip to generate the pattern associated with a given Rule, templates describing the Rule must be specified as data input of the program automatically by the CNN operating system that comes with the chip.
Table 2. All 104 Linearly-Separable Rules, $\kappa=1$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>8</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>16</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>31</td>
<td>32</td>
<td>34</td>
<td>35</td>
<td>42</td>
<td>43</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>55</td>
<td>59</td>
<td>63</td>
<td>64</td>
<td>68</td>
<td>69</td>
<td>76</td>
<td>77</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>81</td>
<td>84</td>
<td>85</td>
<td>87</td>
<td>93</td>
<td>95</td>
<td>112</td>
<td>113</td>
<td>115</td>
<td>117</td>
<td>119</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>136</td>
<td>138</td>
<td>140</td>
<td>142</td>
<td>143</td>
<td>160</td>
<td>162</td>
<td>168</td>
<td>170</td>
<td>171</td>
<td>174</td>
<td>175</td>
<td></td>
</tr>
<tr>
<td>176</td>
<td>178</td>
<td>179</td>
<td>186</td>
<td>187</td>
<td>191</td>
<td>192</td>
<td>196</td>
<td>200</td>
<td>204</td>
<td>205</td>
<td>206</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>212</td>
<td>213</td>
<td>220</td>
<td>221</td>
<td>223</td>
<td>224</td>
<td>232</td>
<td>234</td>
<td>236</td>
<td>238</td>
<td>239</td>
<td>240</td>
<td></td>
</tr>
<tr>
<td>241</td>
<td>242</td>
<td>243</td>
<td>244</td>
<td>245</td>
<td>247</td>
<td>248</td>
<td>250</td>
<td>251</td>
<td>252</td>
<td>253</td>
<td>254</td>
<td>255</td>
<td></td>
</tr>
</tbody>
</table>
Table 3. All $\kappa = 2$ Rules

<table>
<thead>
<tr>
<th>6</th>
<th>9</th>
<th>18</th>
<th>20</th>
<th>22</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>28</th>
<th>30</th>
<th>33</th>
<th>36</th>
<th>37</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>40</td>
<td>41</td>
<td>44</td>
<td>45</td>
<td>52</td>
<td>54</td>
<td>56</td>
<td>57</td>
<td>60</td>
<td>61</td>
<td>62</td>
<td>65</td>
</tr>
<tr>
<td>66</td>
<td>67</td>
<td>70</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>82</td>
<td>86</td>
<td>88</td>
<td>89</td>
<td>90</td>
<td>91</td>
</tr>
<tr>
<td>94</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>106</td>
<td>107</td>
<td>108</td>
</tr>
<tr>
<td>109</td>
<td>110</td>
<td>111</td>
<td>118</td>
<td>120</td>
<td>121</td>
<td>122</td>
<td>123</td>
<td>124</td>
<td>125</td>
<td>126</td>
<td>129</td>
<td>130</td>
</tr>
<tr>
<td>131</td>
<td>132</td>
<td>133</td>
<td>134</td>
<td>135</td>
<td>137</td>
<td>144</td>
<td>145</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>149</td>
<td>151</td>
</tr>
<tr>
<td>152</td>
<td>153</td>
<td>154</td>
<td>155</td>
<td>156</td>
<td>157</td>
<td>158</td>
<td>159</td>
<td>161</td>
<td>164</td>
<td>165</td>
<td>166</td>
<td>167</td>
</tr>
<tr>
<td>169</td>
<td>173</td>
<td>180</td>
<td>181</td>
<td>182</td>
<td>183</td>
<td>185</td>
<td>188</td>
<td>189</td>
<td>190</td>
<td>193</td>
<td>194</td>
<td>195</td>
</tr>
<tr>
<td>198</td>
<td>199</td>
<td>201</td>
<td>203</td>
<td>210</td>
<td>211</td>
<td>214</td>
<td>215</td>
<td>217</td>
<td>218</td>
<td>219</td>
<td>222</td>
<td>225</td>
</tr>
<tr>
<td>227</td>
<td>229</td>
<td>230</td>
<td>231</td>
<td>233</td>
<td>235</td>
<td>237</td>
<td>246</td>
<td>249</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. All $\kappa = 3$ Rules

<table>
<thead>
<tr>
<th>27</th>
<th>29</th>
<th>39</th>
<th>46</th>
<th>53</th>
<th>58</th>
<th>71</th>
<th>78</th>
<th>83</th>
</tr>
</thead>
<tbody>
<tr>
<td>92</td>
<td>105</td>
<td>114</td>
<td>116</td>
<td>139</td>
<td>141</td>
<td>150</td>
<td>163</td>
<td>172</td>
</tr>
<tr>
<td>177</td>
<td>184</td>
<td>197</td>
<td>202</td>
<td>209</td>
<td>216</td>
<td>226</td>
<td>228</td>
<td></td>
</tr>
</tbody>
</table>
E. Templates

Three templates are needed to program a CNN universal chip: the A template, the B template, and the z template (Figure 8). The A template is the feedback term of the CNN nonlinear differential equation. For simplicity and robustness, $a_{00}$ is chosen to be 1, while the other cells of the templates are 0. In the linearly-separable cases, the B and z templates are obtained from the orientation vector [Chua et al, 2002] and [Dogaru & Chua, 1998], or normal vector, and the offset of the single separating plane respectively. These templates also convey the equation that describes the outputs of the corresponding truth table. Usually, the output equation is a signum function, $sgn$, involving analytical variables $u_{i-1}, u_i, u_{i+1}$, since the CNN chip operates analytically. This signum function can be converted into a sign function, $s$, involving symbolic variables $x_{i-1}, x_i, x_{i+1}$ in accordance to symbolic truth tables for better understanding. The signum and sign functions are presented in Figure 7. The separating plane and resulting B and z templates are shown in Figure 8 for Rule 204, which is linearly separable.

![Figure 7. y = sgn (x) and y = s(x), respectively.](image)

$$y = sgn (x)$$

$$y = s (x)$$
Rules that have higher complexity indices are not linearly-separable, but they always can be decomposed into two or three linearly-separable Rules by Boolean Operations (AND, OR). These Rules are also known as *linearly non-separable* Rules. An example is Rule 184, which is equivalent to Rule 186 AND Rule 248 as presented by Figure 9. Table 5 provides a synthesis of all the higher complexity Rules.
<table>
<thead>
<tr>
<th>Rule N</th>
<th>Synthesis using Rules from Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule 6</td>
<td>Rule 7 AND Rule 14</td>
</tr>
<tr>
<td>Rule 9</td>
<td>Rule 11 AND Rule 13</td>
</tr>
<tr>
<td>Rule 18</td>
<td>Rule 19 AND Rule 50</td>
</tr>
<tr>
<td>Rule 20</td>
<td>Rule 21 AND Rule 84</td>
</tr>
<tr>
<td>Rule 22</td>
<td>Rule 23 AND Rule 254</td>
</tr>
<tr>
<td>Rule 24</td>
<td>Rule 31 AND Rule 248</td>
</tr>
<tr>
<td>Rule 25</td>
<td>Rule 59 AND Rule 221</td>
</tr>
<tr>
<td>Rule 26</td>
<td>Rule 31 AND Rule 250</td>
</tr>
<tr>
<td>Rule 27</td>
<td>Rule 31 AND Rule 251</td>
</tr>
<tr>
<td>Rule 28</td>
<td>Rule 31 AND Rule 220</td>
</tr>
<tr>
<td>Rule 29</td>
<td>Rule 31 AND Rule 93</td>
</tr>
<tr>
<td>Rule 30</td>
<td>Rule 31 AND Rule 254</td>
</tr>
<tr>
<td>Rule 33</td>
<td>Rule 35 AND Rule 49</td>
</tr>
<tr>
<td>Rule 36</td>
<td>Rule 47 AND Rule 244</td>
</tr>
<tr>
<td>Rule 37</td>
<td>Rule 47 AND Rule 117</td>
</tr>
<tr>
<td>Rule 38</td>
<td>Rule 55 AND Rule 238</td>
</tr>
<tr>
<td>Rule 39</td>
<td>Rule 47 AND Rule 55</td>
</tr>
<tr>
<td>Rule 40</td>
<td>Rule 42 AND Rule 168</td>
</tr>
<tr>
<td>Rule 41</td>
<td>Rule 43 AND Rule 253</td>
</tr>
<tr>
<td>Rule 44</td>
<td>Rule 47 AND Rule 236</td>
</tr>
<tr>
<td>Rule 45</td>
<td>Rule 47 AND Rule 253</td>
</tr>
<tr>
<td>Rule 46</td>
<td>Rule 47 AND Rule 254</td>
</tr>
<tr>
<td>Rule 52</td>
<td>Rule 55 AND Rule 244</td>
</tr>
<tr>
<td>Rule 53</td>
<td>Rule 55 AND Rule 245</td>
</tr>
<tr>
<td>Rule 54</td>
<td>Rule 55 AND Rule 254</td>
</tr>
<tr>
<td>Rule 56</td>
<td>Rule 59 AND Rule 248</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rule N</th>
<th>Synthesis using Rules from Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule 57</td>
<td>Rule 59 AND Rule 253</td>
</tr>
<tr>
<td>Rule 58</td>
<td>Rule 59 AND Rule 254</td>
</tr>
<tr>
<td>Rule 60</td>
<td>Rule 63 AND Rule 252</td>
</tr>
<tr>
<td>Rule 61</td>
<td>Rule 63 AND Rule 253</td>
</tr>
<tr>
<td>Rule 62</td>
<td>Rule 63 AND Rule 254</td>
</tr>
<tr>
<td>Rule 65</td>
<td>Rule 69 AND Rule 81</td>
</tr>
<tr>
<td>Rule 66</td>
<td>Rule 79 AND Rule 242</td>
</tr>
<tr>
<td>Rule 67</td>
<td>Rule 79 AND Rule 243</td>
</tr>
<tr>
<td>Rule 70</td>
<td>Rule 87 AND Rule 206</td>
</tr>
<tr>
<td>Rule 71</td>
<td>Rule 79 AND Rule 87</td>
</tr>
<tr>
<td>Rule 72</td>
<td>Rule 76 AND Rule 232</td>
</tr>
<tr>
<td>Rule 73</td>
<td>Rule 77 AND Rule 251</td>
</tr>
<tr>
<td>Rule 74</td>
<td>Rule 79 AND Rule 234</td>
</tr>
<tr>
<td>Rule 75</td>
<td>Rule 79 AND Rule 251</td>
</tr>
<tr>
<td>Rule 78</td>
<td>Rule 79 AND Rule 254</td>
</tr>
<tr>
<td>Rule 82</td>
<td>Rule 87 AND Rule 250</td>
</tr>
<tr>
<td>Rule 83</td>
<td>Rule 87 AND Rule 243</td>
</tr>
<tr>
<td>Rule 86</td>
<td>Rule 87 AND Rule 254</td>
</tr>
<tr>
<td>Rule 88</td>
<td>Rule 93 AND Rule 248</td>
</tr>
<tr>
<td>Rule 89</td>
<td>Rule 93 AND Rule 251</td>
</tr>
<tr>
<td>Rule 90</td>
<td>Rule 95 AND Rule 250</td>
</tr>
<tr>
<td>Rule 91</td>
<td>Rule 95 AND Rule 251</td>
</tr>
<tr>
<td>Rule 92</td>
<td>Rule 95 AND Rule 220</td>
</tr>
<tr>
<td>Rule 94</td>
<td>Rule 95 AND Rule 254</td>
</tr>
<tr>
<td>Rule 96</td>
<td>Rule 112 AND Rule 224</td>
</tr>
<tr>
<td>Rule 97</td>
<td>Rule 113 AND Rule 239</td>
</tr>
<tr>
<td>Rule N</td>
<td>Synthesis using Rules from Table 1</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Rule 98</td>
<td>Rule 115 AND Rule 234</td>
</tr>
<tr>
<td>Rule 99</td>
<td>Rule 115 AND Rule 239</td>
</tr>
<tr>
<td>Rule 100</td>
<td>Rule 117 AND Rule 238</td>
</tr>
<tr>
<td>Rule 101</td>
<td>Rule 117 AND Rule 239</td>
</tr>
<tr>
<td>Rule 102</td>
<td>Rule 119 AND Rule 238</td>
</tr>
<tr>
<td>Rule 103</td>
<td>Rule 119 AND Rule 239</td>
</tr>
<tr>
<td>Rule 104</td>
<td>Rule 127 AND Rule 232</td>
</tr>
<tr>
<td>Rule 105</td>
<td>(Rule 43 OR Rule 64) AND Rule 253</td>
</tr>
<tr>
<td>Rule 106</td>
<td>Rule 127 AND Rule 234</td>
</tr>
<tr>
<td>Rule 107</td>
<td>Rule 43 OR Rule 64</td>
</tr>
<tr>
<td>Rule 108</td>
<td>Rule 127 AND Rule 236</td>
</tr>
<tr>
<td>Rule 109</td>
<td>Rule 32 OR Rule 77</td>
</tr>
<tr>
<td>Rule 110</td>
<td>Rule 127 AND Rule 238</td>
</tr>
<tr>
<td>Rule 111</td>
<td>Rule 127 AND Rule 239</td>
</tr>
<tr>
<td>Rule 114</td>
<td>Rule 115 AND Rule 242</td>
</tr>
<tr>
<td>Rule 116</td>
<td>Rule 117 AND Rule 252</td>
</tr>
<tr>
<td>Rule 118</td>
<td>Rule 119 AND Rule 254</td>
</tr>
<tr>
<td>Rule 120</td>
<td>Rule 127 AND Rule 248</td>
</tr>
<tr>
<td>Rule 121</td>
<td>Rule 8 OR Rule 113</td>
</tr>
<tr>
<td>Rule 122</td>
<td>Rule 127 AND Rule 250</td>
</tr>
<tr>
<td>Rule 123</td>
<td>Rule 127 AND Rule 251</td>
</tr>
<tr>
<td>Rule 124</td>
<td>Rule 127 AND Rule 252</td>
</tr>
<tr>
<td>Rule 125</td>
<td>Rule 127 AND Rule 253</td>
</tr>
<tr>
<td>Rule 126</td>
<td>Rule 127 AND Rule 254</td>
</tr>
<tr>
<td>Rule 129</td>
<td>Rule 143 AND Rule 241</td>
</tr>
<tr>
<td>Rule 130</td>
<td>Rule 138 AND Rule 162</td>
</tr>
</tbody>
</table>
Therefore, instead of finding the templates for all 256 1-D CA Rules, it is sufficient to implement the templates of the 104 linearly-separable Rules to program the CNN universal chip.

<table>
<thead>
<tr>
<th>Rule N</th>
<th>Synthesis using Rules from Table 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule 164</td>
<td>Rule 174 AND Rule 244</td>
</tr>
<tr>
<td>Rule 165</td>
<td>Rule 175 AND Rule 245</td>
</tr>
<tr>
<td>Rule 166</td>
<td>Rule 174 AND Rule 247</td>
</tr>
<tr>
<td>Rule 167</td>
<td>Rule 175 AND Rule 247</td>
</tr>
<tr>
<td>Rule 169</td>
<td>Rule 171 AND Rule 253</td>
</tr>
<tr>
<td>Rule 172</td>
<td>Rule 174 AND Rule 253</td>
</tr>
<tr>
<td>Rule 173</td>
<td>Rule 175 AND Rule 253</td>
</tr>
<tr>
<td>Rule 177</td>
<td>Rule 179 AND Rule 241</td>
</tr>
<tr>
<td>Rule 180</td>
<td>Rule 191 AND Rule 244</td>
</tr>
<tr>
<td>Rule 181</td>
<td>Rule 191 AND Rule 245</td>
</tr>
<tr>
<td>Rule 182</td>
<td>Rule 4 OR Rule 178</td>
</tr>
<tr>
<td>Rule 183</td>
<td>Rule 191 AND Rule 247</td>
</tr>
<tr>
<td>Rule 184</td>
<td>Rule 186 AND Rule 248</td>
</tr>
<tr>
<td>Rule 185</td>
<td>Rule 187 AND Rule 253</td>
</tr>
<tr>
<td>Rule 188</td>
<td>Rule 191 AND Rule 252</td>
</tr>
<tr>
<td>Rule 189</td>
<td>Rule 191 AND Rule 253</td>
</tr>
<tr>
<td>Rule 190</td>
<td>Rule 191 AND Rule 254</td>
</tr>
<tr>
<td>Rule 193</td>
<td>Rule 205 AND Rule 241</td>
</tr>
<tr>
<td>Rule 194</td>
<td>Rule 206 AND Rule 242</td>
</tr>
<tr>
<td>Rule 195</td>
<td>Rule 207 AND Rule 243</td>
</tr>
<tr>
<td>Rule 197</td>
<td>Rule 205 AND Rule 213</td>
</tr>
<tr>
<td>Rule 198</td>
<td>Rule 206 AND Rule 247</td>
</tr>
<tr>
<td>Rule 199</td>
<td>Rule 207 AND Rule 247</td>
</tr>
<tr>
<td>Rule 201</td>
<td>Rule 205 AND Rule 251</td>
</tr>
<tr>
<td>Rule 202</td>
<td>Rule 206 AND Rule 251</td>
</tr>
<tr>
<td>Rule 203</td>
<td>Rule 207 AND Rule 251</td>
</tr>
</tbody>
</table>
F. Optimal Templates

Mathematically speaking, there are an infinite number of possible separating planes. Since the B and z templates are based on the normal vector and the offset of the separating plane, respectively, there are also an infinite number of possible templates per Boolean Rule. This is easily seen in the example of Rule 16, shown in Figure 11.

To provide a basis for studies concerning CNN universal chips and Boolean Rules, it is best to adhere to a standard set of templates. Among the infinite template possibilities, there is only one set of optimal CNN templates. In this paper, the optimal separating plane is defined as the plane that is at the maximum projected distance possible from each of the cube’s 8...
vertices, but which still separates the red vertices from the blue ones. The corresponding optimal CNN templates are derived from the normal vectors and offsets of these optimal separating planes. An algorithm to find this plane is provided in the Appendix. In short, a plane is selected that satisfactorily separates the red vertices from the blue. The projected distance from each vertex to the plane is found, and the minimum distance is maximized by adjusting the plane until an optimal plane is achieved.

2 Gallery of Templates
The optimal CNN templates of the 104 linearly-separable 1-D CA are implemented and presented in table 6. The truth table, Cube, templates, and output formula are presented for each of the 104 Boolean Rules. The truth tables and output formula use symbolic variables $x_{i-1}, x_i, x_{i+1}$ for better understanding, but the templates are found based on analytic variables $u_{i-1}, u_i, u_{i+1}$ since the CNN universal chip operates on a numerical analytical basis.
Table 6. A Gallery of Templates of all 104 Linearly Separable CA

<table>
<thead>
<tr>
<th>Rule</th>
<th>Template</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td><img src="image" alt="Rule 0 Template" /></td>
<td>$y = -1$</td>
</tr>
<tr>
<td>1</td>
<td><img src="image" alt="Rule 1 Template" /></td>
<td>$y = \phi[-2x_{i,1} - 2x_1 + 2x_{i,1} + 1]$</td>
</tr>
<tr>
<td>2</td>
<td><img src="image" alt="Rule 2 Template" /></td>
<td>$y = \phi[-2x_{i,1} - 2x_1 + 2x_{i,1} + 1]$</td>
</tr>
<tr>
<td>3</td>
<td><img src="image" alt="Rule 3 Template" /></td>
<td>$y = \phi[-2x_{i,1} - 2x_1 + 1]$</td>
</tr>
</tbody>
</table>
Rule 4 \( \kappa = 1 \)

Template

\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -1 \end{bmatrix} \quad z = -2
\]

Formula

\[ y = \phi \left[ -2x_{i+1} + 2x_i - 2x_{i+1} - 1 \right] \]

Rule 5 \( \kappa = 1 \)

Template

\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = -1
\]

Formula

\[ y = \phi \left[ -2x_{i+1} - 2x_{i+1} + 1 \right] \]

Rule 7 \( \kappa = 1 \)

Template

\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -1 \end{bmatrix} \quad z = -1
\]

Formula

\[ y = \phi \left[ -4x_{i+1} - 2x_i - 2x_{i+1} + 3 \right] \]

Rule 8 \( \kappa = 1 \)

Template

\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 1 & 1 \end{bmatrix} \quad z = -2
\]

Formula

\[ y = \phi \left[ -2x_{i+1} + 2x_i + 2x_{i+1} - 3 \right] \]
Rule 10 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad z = -1
\]

Formula
\[ y = a \{-2x_{i-1} + 2x_{i,1} - 1\} \]

Rule 11 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad z = -1
\]

Formula
\[ y = a \{-4x_i - 2x_i + 2x_{i+1} + 1\} \]

Rule 12 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad z = -1
\]

Formula
\[ y = a \{-2x_{i,1} + 2x_i - 1\} \]

Rule 13 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & -1 & 1 \end{bmatrix}, \quad z = -1
\]

Formula
\[ y = a \{-4x_{i,1} + 2x_i - 2x_{i+1} + 1\} \]
**Rule 14**

Template:

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -2 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = -1 \]

Formula:

\[ y = a \left( -4x_{i-1} + 2x_i + 2x_{i+1} - 1 \right) \]

**Rule 15**

Template:

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \]

\[ z = 0 \]

Formula:

\[ y = a \left( -2x_i + 1 \right) \]

**Rule 16**

Template:

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -2 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = -2 \]

Formula:

\[ y = a \left( 2x_{i-1} - 2x_i - 2x_{i+1} - 1 \right) \]

**Rule 17**

Template:

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = -1 \]

Formula:

\[ y = a \left( -2x_i - 2x_{i+1} + 1 \right) \]
**Rule 19**

\[ \kappa = 1 \]

Template:
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad z = -1 \]

Formula:
\[ y = \sigma [-2x_{i,1} - 4x_i - 2x_{i+1} + 3] \]

---

**Rule 21**

\[ \kappa = 1 \]

Template:
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad z = -1 \]

Formula:
\[ y = \sigma [-2x_i - 2x_{i+1} + 4x_{i+1} + 3] \]

---

**Rule 23**

\[ \kappa = 1 \]

Template:
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 0 \]

Formula:
\[ y = \sigma [-2x_{i,1} - 2x_i - 2x_{i+1} + 3] \]

---

**Rule 31**

\[ \kappa = 1 \]

Template:
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & -1 & -1 \end{bmatrix} \quad z = -1 \]

Formula:
\[ y = \sigma [-4x_{i,1} - 2x_i - 2x_{i+1} + 5] \]
Rule 32

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

Formula

\[ y = a \left[ 2x_{i,1} - 2x_i + 2x_{i+1} - 3 \right] \]

Rule 34

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

Formula

\[ y = a \left[ -2x_i + 2x_{i+1} - 1 \right] \]

Rule 35

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

Formula

\[ y = a \left[ -2x_{i,1} - 4x_i + 2x_{i+1} + 1 \right] \]

Rule 42

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ -1 & -2 & 2 \\ 0 & 0 & 0 \end{bmatrix} \]

Formula

\[ y = a \left[ -2x_{i,1} - 2x_i + 4x_{i+1} - 1 \right] \]
Rule 43  $\kappa = 1$

Template:

\[
A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & -1 \\
0 & 0 & 0 \end{bmatrix} \quad z = 0
\]

Formula:

\[
y = a \left[ -2x_{i,1} - 2x_i + 2x_{i+1} + 1 \right]
\]

Rule 47  $\kappa = 1$

Template:

\[
A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 1 & 0 \end{bmatrix} \quad z = 1
\]

Formula:

\[
y = a \left[ -4x_{i,1} - 2x_i + 2x_{i+1} + 3 \right]
\]

Rule 48  $\kappa = 1$

Template:

\[
A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad z = -1
\]

Formula:

\[
y = a \left[ 2x_{i,1} - 2x_i - 1 \right]
\]

Rule 49  $\kappa = 1$

Template:

\[
A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad z = -1
\]

Formula:

\[
y = a \left[ 2x_{i,1} - 4x_i - 2x_{i+1} + 1 \right]
\]
Rule 50

\[ x_{i+1}, x_i, x_{i+1}, y_i \]

| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix}, z = -1 \]

Formula

\[ y = \sigma [ 2x_{i+1} - 4x_i + 2x_{i+1} - 1 ] \]

Rule 51

\[ x_{i+1}, x_i, x_{i+1}, y_i \]

| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, z = 0 \]

Formula

\[ y = \sigma [ -2x_i + 1 ] \]

Rule 55

\[ x_{i+1}, x_i, x_{i+1}, y_i \]

| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 1 & -2 \\ 0 & 0 \end{bmatrix}, z = -1 \]

Formula

\[ y = \sigma [ -2x_{i+1} - 4x_i - 2x_{i+1} + 5 ] \]

Rule 59

\[ x_{i+1}, x_i, x_{i+1}, y_i \]

| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, z = -1 \]

Formula

\[ y = \sigma [ -2x_{i+1} - 4x_i + 2x_{i+1} + 3 ] \]
Rule 63 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = a \cdot [-2x_{i-1} - 2x_i + 3] \]

Rule 64 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = -2 \]

Formula

\[ y = a \cdot [2x_{i-1} + 2x_i - 2x_{i+1} + 3] \]

Rule 68 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = -1 \]

Formula

\[ y = a \cdot [2x_i \cdot 2x_{i+1} - 1] \]

Rule 69 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & -2 \\ 0 & 0 & 0 \end{bmatrix} \quad z = -1 \]

Formula

\[ y = a \cdot [-2x_{i-1} + 2x_i - 4x_{i+1} + 1] \]
**Rule 81**

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
\sigma_X & \sigma_i & \sigma_{i+1} & \sigma_Y \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 \\
\end{array}
\]

Template:

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix},
B = \begin{bmatrix}
0 & 0 & 0 \\
1 & -1 & -2 \\
0 & 0 & 0
\end{bmatrix}
\]

Formula:

\[ y = \sigma \left( 2x_{i+1} - 2x_i - 4x_{i+1} + 1 \right) \]

**Rule 84**

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
\sigma_X & \sigma_i & \sigma_{i+1} & \sigma_Y \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 \\
1 & 1 & 1 & 0 \\
1 & 1 & 1 & 0 \\
\end{array}
\]

Template:

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix},
B = \begin{bmatrix}
0 & 0 & 0 \\
1 & 1 & -2 \\
0 & 0 & 0
\end{bmatrix}
\]

Formula:

\[ y = \sigma \left( 2x_{i+1} + 2x_i - 4x_{i+1} - 1 \right) \]

**Rule 85**

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
\sigma_X & \sigma_i & \sigma_{i+1} & \sigma_Y \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 \\
\end{array}
\]

Template:

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix},
B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

Formula:

\[ y = \sigma \left( -2x_{i+1} + 1 \right) \]

**Rule 87**

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
\sigma_X & \sigma_i & \sigma_{i+1} & \sigma_Y \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 \\
\end{array}
\]

Template:

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix},
B = \begin{bmatrix}
0 & 0 & 0 \\
-1 & -1 & -2 \\
0 & 0 & 0
\end{bmatrix}
\]

Formula:

\[ y = \sigma \left( -2x_{i+1} - 2x_i - 4x_{i+1} + 5 \right) \]
**Rule 93**

\[ z = 1 \]

**Template**

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} \quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
-1 & -1 & -2 \\
\end{bmatrix}
\]

**Formula**

\[ y = a \left( -2x_{i-1} + 2x_i - 4x_{i+1} + 3 \right) \]

---

**Rule 95**

\[ z = 1 \]

**Template**

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
-1 & 0 & 1 \\
\end{bmatrix} \quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

**Formula**

\[ y = a \left( -2x_{i-1} - 2x_{i+1} + 3 \right) \]

---

**Rule 112**

\[ z = -1 \]

**Template**

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} \quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
-1 & 1 & -1 \\
\end{bmatrix}
\]

**Formula**

\[ y = a \left( 4x_{i-1} - 2x_i - 2x_{i+1} - 1 \right) \]

---

**Rule 113**

\[ z = 0 \]

**Template**

\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix} \quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

**Formula**

\[ y = a \left( 2x_{i-1} + 2x_i - 2x_{i+1} + 1 \right) \]
Rule 115 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & -1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \phi [ 2x_{i+1} - 4x_i - 2x_{i+1} + 3 ] \]

Rule 117 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \phi [ 2x_{i+1} - 2x_i - 4x_{i+1} + 3 ] \]

Rule 119 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \phi [ 2x_{i+1} - 2x_i + 3 ] \]

Rule 127 \( \kappa = 1 \)

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 2 \]

Formula

\[ y = \phi [ -2x_{i+1} - 2x_i - 2x_{i+1} + 5 ] \]
Rule 142 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]

Formula
\[
y = \phi \left( -2x_{i-1} + 2x_i + 2x_{i+1} - 1 \right)
\]

Rule 143 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}
\]

Formula
\[
y = \phi \left( -4x_{i-1} + 2x_i + 2x_{i+1} + 1 \right)
\]

Rule 160 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}
\]

Formula
\[
y = \phi \left( 2x_{i-1} + 2x_{i+1} - 3 \right)
\]

Rule 162 \( \kappa = 1 \)

Template
\[
A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}
\]

Formula
\[
y = \phi \left( 2x_{i+1} - 2x_i + 4x_{i+1} - 3 \right)
\]
Rule 168 \( \kappa = 1 \)

**Template**

\[
A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}, \quad z = -1
\]

**Formula**

\( y = \sigma_1 (2x_{i+1} + 2x_i + 4x_{i+1} - 5) \)

Rule 170 \( \kappa = 1 \)

**Template**

\[
A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \quad z = 0
\]

**Formula**

\( y = \sigma_1 (2x_{i+1} - 1) \)

Rule 171 \( \kappa = 1 \)

**Template**

\[
A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}, \quad z = -1
\]

**Formula**

\( y = \sigma_1 (-2x_{i+1} - 2x_i + 4x_{i+1} + 1) \)

Rule 174 \( \kappa = 1 \)

**Template**

\[
A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}, \quad z = -1
\]

**Formula**

\( y = \sigma_1 (-2x_{i+1} + 2x_i + 4x_{i+1} - 1) \)
Rule 186

\[ x_{i+1} x_i x_{i+1} y_i \]
\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template:
\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad B = \begin{bmatrix}
0 & 0 & 0 \\
1 & -1 & 2 \\
0 & 0 & 0
\end{bmatrix}
\quad z = 1
\]

Formula:
\[ y = \mathcal{A} [2x_{i+1} - 2x_i + 4x_{i+1} - 1] \]

Rule 187

\[ x_{i+1} x_i x_{i+1} y_i \]
\[
\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template:
\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad z = 1
\]

Formula:
\[ y = \mathcal{A} [-2x_i + 2x_{i+1} + 1] \]

Rule 191

\[ x_{i+1} x_i x_{i+1} y_i \]
\[
\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template:
\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad B = \begin{bmatrix}
0 & 0 & 0 \\
1 & -1 & 2 \\
0 & 0 & 0
\end{bmatrix}
\quad z = 1
\]

Formula:
\[ y = \mathcal{A} [-2x_i + 2x_{i+1} + 3] \]

Rule 192

\[ x_{i+1} x_i x_{i+1} y_i \]
\[
\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template:
\[
A = \begin{bmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad B = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 0
\end{bmatrix}
\quad z = 1
\]

Formula:
\[ y = \mathcal{A} [2x_{i+1} + 2x_i - 3] \]
Rule 196  \( \kappa = 1 \)

Template
\[
\begin{align*}
A = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
B = & \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 0 & 0 & 0 \end{pmatrix} \\
z = & -1
\end{align*}
\]

Formula
\[ y = \sigma \left( 2x_{i+1} + 4x_i - 2x_{i+1} - 3 \right) \]

Rule 200  \( \kappa = 1 \)

Template
\[
\begin{align*}
A = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
B = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
z = & -1
\end{align*}
\]

Formula
\[ y = \sigma \left( 2x_{i+1} + 4x_i - 2x_{i+1} - 5 \right) \]

Rule 204  \( \kappa = 1 \)

Template
\[
\begin{align*}
A = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
B = & \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 0 & 0 & 0 \end{pmatrix} \\
z = & 0
\end{align*}
\]

Formula
\[ y = \sigma \left( 2x_i \cdot 1 \right) \]

Rule 205  \( \kappa = 1 \)

Template
\[
\begin{align*}
A = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
B = & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
z = & 1
\end{align*}
\]

Formula
\[ y = \sigma \left( -2x_{i+1} + 4x_i - 2x_{i+1} + 1 \right) \]
**Rule 206**  \( \kappa = 1 \)

**Template**

\[
\begin{bmatrix}
A = & 0 & 0 & 0 \\
& 1 & 0 & 0 \\
& 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
B = & 0 & 0 & 0 \\
& 2 & 1 & -1 \\
& 0 & 0 & 0
\end{bmatrix}
\]

**Formula**

\[
y = \phi[-2x_{i-1} + 4x_i + 2x_{i+1} - 1]
\]

**Rule 207**  \( \kappa = 1 \)

**Template**

\[
\begin{bmatrix}
A = & 0 & 0 & 0 \\
& 0 & 0 & 1 \\
& 0 & 1 & 0 \\
& 0 & 1 & 1 \\
& 1 & 0 & 0 \\
& 1 & 0 & 1 \\
& 1 & 1 & 0 \\
& 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
B = & 0 & 0 & 0 \\
& 0 & 1 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0
\end{bmatrix}
\]

**Formula**

\[
y = \phi[-2x_{i-1} + 2x_i + 1]
\]

**Rule 208**  \( \kappa = 1 \)

**Template**

\[
\begin{bmatrix}
A = & 0 & 0 & 0 \\
& 1 & 0 & 0 \\
& 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
B = & 0 & 0 & 0 \\
& 0 & 0 & 1 \\
& 0 & 0 & 1 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0
\end{bmatrix}
\]

**Formula**

\[
y = \phi[4x_{i-1} + 2x_i - 2x_{i+1} - 3]
\]

**Rule 212**  \( \kappa = 1 \)

**Template**

\[
\begin{bmatrix}
A = & 0 & 0 & 0 \\
& 0 & 0 & 1 \\
& 0 & 1 & 0 \\
& 0 & 1 & 1 \\
& 1 & 0 & 0 \\
& 1 & 0 & 1 \\
& 1 & 1 & 0 \\
& 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
B = & 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0 \\
& 0 & 0 & 0
\end{bmatrix}
\]

**Formula**

\[
y = \phi[2x_{i-1} + 2x_i - 2x_{i+1} - 1]
\]
Rule 213

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
X_i & X_{i+1} & Y_i & \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \sigma [2x_{i+1} + 2x_i - 4x_{i+1} + 1] \]

Rule 220

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
X_i & X_{i+1} & Y_i & \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 1 \\
0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \sigma [2x_{i+1} + 4x_i - 2x_{i+1} - 1] \]

Rule 221

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
X_i & X_{i+1} & Y_i & \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & -1 \\
0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula

\[ y = \sigma [2x_i - 2x_{i+1} + 1] \]

Rule 223

\[ \kappa = 1 \]

\[
\begin{array}{cccc}
X_i & X_{i+1} & Y_i & \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}
\]

Template

\[ A = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\
0 & 1 & -1 \\
0 & 0 & 0 \end{bmatrix} \quad z = 2 \]

Formula

\[ y = \sigma [-2x_{i+1} + 2x_i - 2x_{i+1} + 3] \]
Rule 224

Template: $A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $z = -1$

Formula: $y = \sigma(4x_{i+1} + 2x_i + 2x_{i-1} - 5)$

Rule 232

Template: $A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $z = 0$

Formula: $y = \sigma(2x_{i+1} + 2x_i + 2x_{i-1} - 3)$

Rule 234

Template: $A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $z = -1$

Formula: $y = \sigma(2x_{i+1} + 2x_i + 4x_{i-1} - 3)$

Rule 236

Template: $A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $z = -1$

Formula: $y = \sigma(2x_{i+1} + 4x_i + 2x_{i-1} - 3)$
**Rule 238**  
\[ \kappa = 1 \]

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = 1 \]

Formula
\[ y = \sigma [ 2x_i + 2x_{i+1} - 1 ] \]

**Rule 239**  
\[ \kappa = 1 \]

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = 1 \]

Formula
\[ y = \sigma [ -2x_{i-1} + 2x_i + 2x_{i+1} + 1 ] \]

**Rule 240**  
\[ \kappa = 1 \]

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = 0 \]

Formula
\[ y = \sigma [ 2x_{i-1} - 1 ] \]

**Rule 241**  
\[ \kappa = 1 \]

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \]

\[ z = -1 \]

Formula
\[ y = \sigma [ 4x_{i+1} - 2x_i - 2x_{i-1} + 1 ] \]
<table>
<thead>
<tr>
<th>Rule 242</th>
<th>Rule 243</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa = 1$</td>
<td>$\kappa = 1$</td>
</tr>
</tbody>
</table>

**Template**

$$ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 1 & -1 \end{bmatrix} \quad z = 1 $$

**Formula**

$$ y = \delta \left[ 4x_{i+1} - 2x_i + 2x_{i+1} - 1 \right] $$

<table>
<thead>
<tr>
<th>Rule 244</th>
<th>Rule 245</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\kappa = 1$</td>
<td>$\kappa = 1$</td>
</tr>
</tbody>
</table>

**Template**

$$ A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 $$

**Formula**

$$ y = \delta \left[ 2x_{i+1} - 2x_i + 1 \right] $$
Rule 247

Template
$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$

$\kappa = 1$

$x_{i+1} \quad x_i \quad x_{i+1} \quad y_i$

\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}

Formula
$y = \phi [2x_{i+1} - 2x_i - 2x_{i+1} + 3]$

Rule 248

Template
$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$

$\kappa = 1$

$x_{i+1} \quad x_i \quad x_{i+1} \quad y_i$

\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}

Formula
$y = \phi [4x_{i+1} + 2x_i + 2x_{i+1} - 3]$

Rule 250

Template
$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$

$\kappa = 1$

$x_{i+1} \quad x_i \quad x_{i+1} \quad y_i$

\begin{array}{cccc}
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}

Formula
$y = \phi [2x_{i+1} + 2x_i - 1]$

Rule 251

Template
$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$

$\kappa = 1$

$x_{i+1} \quad x_i \quad x_{i+1} \quad y_i$

\begin{array}{cccc}
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 \\
\end{array}

Formula
$y = \phi [2x_{i+1} - 2x_i + 2x_{i+1} + 1]$
Rule 252

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula
\[ y = a \left\{ 2x_{i+1} + 2x_i - 1 \right\} \]

Rule 253

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 2 \]

Formula
\[ y = a \left\{ 2x_{i+1} + 2x_i - 2x_{i+1} + 1 \right\} \]

Rule 254

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 2 \]

Formula
\[ y = a \left\{ 2x_{i+1} + 2x_i + 2x_{i+1} - 1 \right\} \]

Rule 255

Template
\[ A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad z = 1 \]

Formula
\[ y = 1 \]
3 Concluding Remarks

After compiling a library of optimal CNN templates for linearly-separable 1-D CA, a next step would be to optimize the current library for 2-D CA. Interested readers may browse the Appendix to use the algorithm to implement the optimal CNN templates for linearly-separable 1-D CA, and then move on to investigating 2-D CA. In the process of compiling the optimal CNN templates for linearly-separable 1-D CA, differences between the optimal CNN templates and those provided in [Chua, et al, 2002] may be noted. In particular, the templates for Rule 63 are incorrect, and should be revised as follows: \([b_1, b_2, b_3]\) should equal to \([-1, -1, 0]\) instead of \([0, -1, 0]\). To check the validity of the templates generated by the algorithm, the CANDY simulator was run and the dynamic firing patterns were compared.

4 Acknowledgements

Many thanks to Ram Rajagopal for helping with SVM concepts and Akos Zarandi for showing us how to use CANDY to simulate CA.

References


Schwaighofer, A. [2002] SVM toolbox for MATLAB.

http://ida.first.fraunhofer.de/~anton/software.html

Appendix

In order to find the optimal separating plane, we use the concept of Support Vector Machines [Moore, 2007]. Often we are interested in classifying data. These data points may not necessarily be points in $\mathbb{R}^2$ but may be multidimensional $\mathbb{R}^n$ points. We are interested in whether we can separate them by a n-1 dimensional hyperplane. This is a typical form of linear classifier. There are many linear classifiers that might satisfy this property. However, we are additionally interested to find out if we can achieve maximum separation (margin) between the two classes. Now, if such a hyperplane exists, the hyperplane is clearly of interest and is known as the maximum-margin hyperplane.

Consider Figure 12. The goal is to separate the "x"s from the "o"s using a hyperplane that is at maximum distance between the two classes. We can consider the data points to be of the form:

$\{(x_1, c_1), (x_2, c_2), \ldots, (x_n, c_n)\}$

Figure 12. Maximum-margin hyperplanes for a SVM trained with samples from two classes. Samples along the hyperplanes are called support vectors

---

In our case, the "x"s could be the red vertices and the "o"s could be the blue vertices. We have thus colored the "x"s and "o"s that would give rise to the support vectors.
Here the $c_i$ is either 1 or -1. This constant denotes the class to which point $x_i$ belongs to (for instance, if the point is an “x” then $c_i$ is 1 and if the point is a “o” then $c_i$ is -1). Each $x_i$ is an n-dimensional real vector, usually of scaled $[0,1]$ or $[-1,1]$ values. Now, the dividing hyperplane takes the form:

$$w \cdot x - b = 0$$

The vector $w$ points perpendicular to the separating hyperplane. Adding the offset parameter $b$ allows us to increase the margin. In its absence, the hyperplane is forced to pass through the origin, restricting the solution. As we are interested in the maximum margin, we are interested in the support vectors and the parallel hyperplanes closest to these support vectors in either class, refer to Figure 12. It can be shown that these hyperplanes can be described by equations:

$$w \cdot x - b = 1,$$
$$w \cdot x - b = -1$$

In our case, the points are linearly separable. Therefore we can select the hyperplanes so that there are no points between them and then try to maximize their distance. By using geometry, we find the distance between the hyperplanes is $\frac{2}{|w|}$, so we want to minimize $|w|$. To exclude data points, we need to ensure that for all $i$ either:

$$w \cdot x_i - b \geq 1, \quad \text{or} \quad w \cdot x_i - b \leq -1$$

This can be rewritten as:

$$c_i (w \cdot x_i - b) \geq 1, \quad 1 \leq i \leq n \quad \text{(1)}$$

The problem now is to minimize $|w|$ subject to the constraint in (1). That is:

**Minimize** $|w|$ **subject to** $c_i (w \cdot x_i - b) \geq 1, \quad 1 \leq i \leq n$

The equation above can be solved using a mathematical package. We use the MATLAB toolbox from [Schwaighofer, 2002] (you need to have the Optimization toolbox from Mathworks). First download and unzip the toolbox from [Schwaighofer, 2002]. Here are MATLAB commands to find the normal vector and the offset of the optimal-separating plane for Rule 95. The first line sets up the input vector, the second line the output vector corresponding to rule 95. The last two lines use the SVM toolbox to obtain the normal vector $w$ and the offset $b$ for the optimal separating plane.
>> U = [-1 -1 -1; -1 -1 1; -1 1 -1; -1 1 1; 1 -1 -1; 1 -1 1; 1 1 -1; 1 1 1];
>> Y95 = [1; 1; 1; 1; 1; -1; 1; -1];
>> net_setup = svm(3,'linear',[],10);
>> net95 = svmtrain(net_setup,U,Y95)