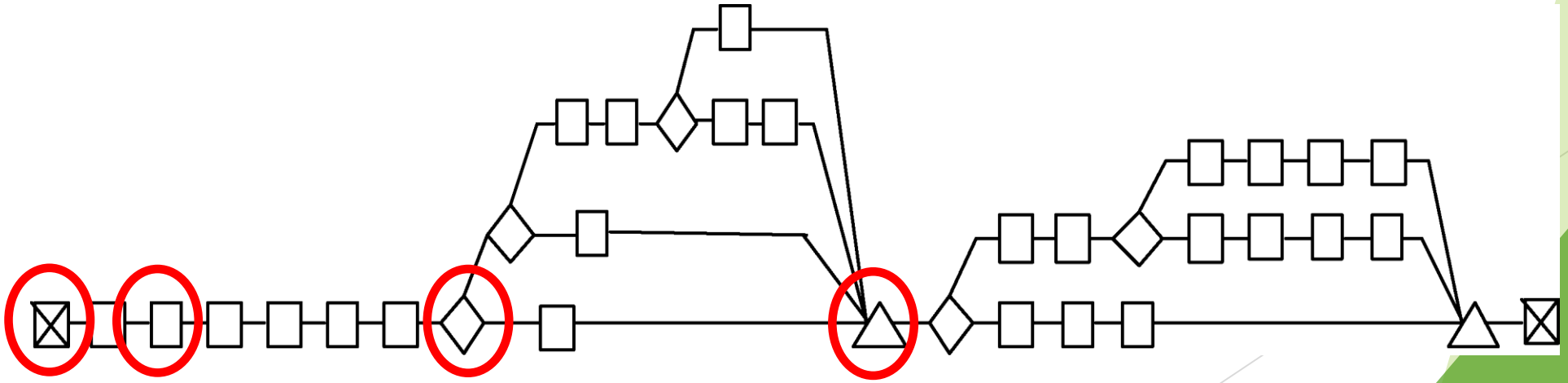
静态结构分析（下）

函数控制流图

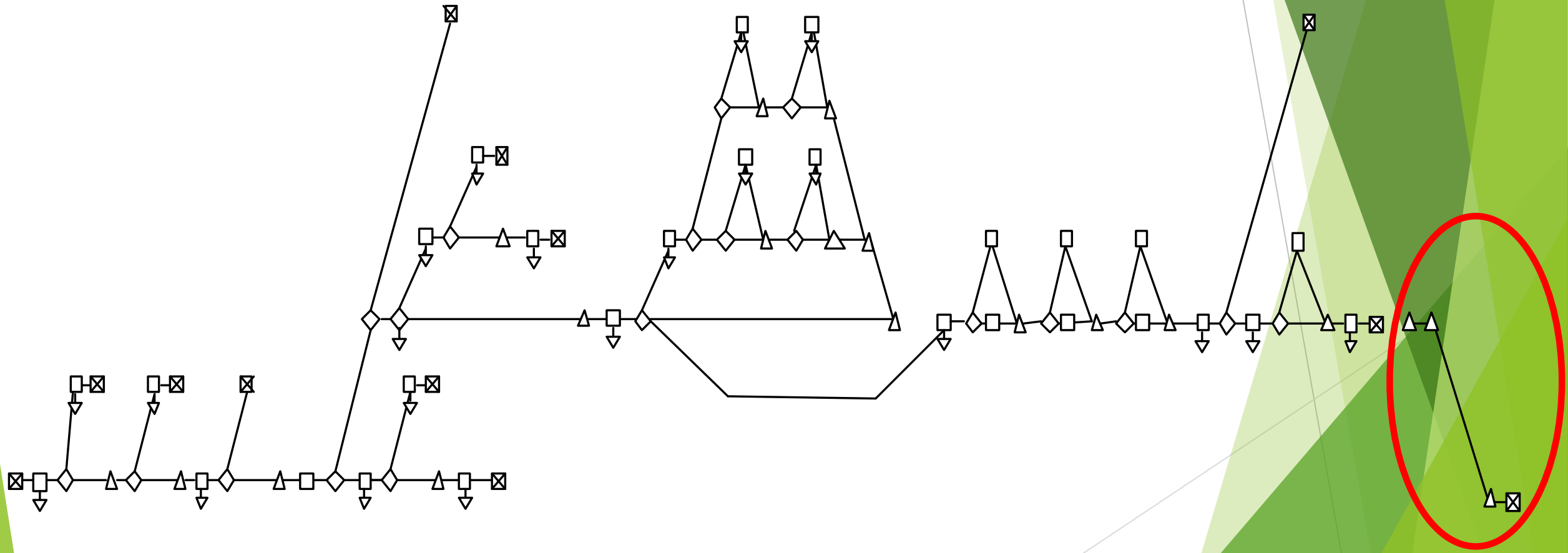
- 由节点和边组成的有向图
  - 节点表示一条或多条语句
  - 边表示节点之间的控制走向，即语句的执行



函数控制流图

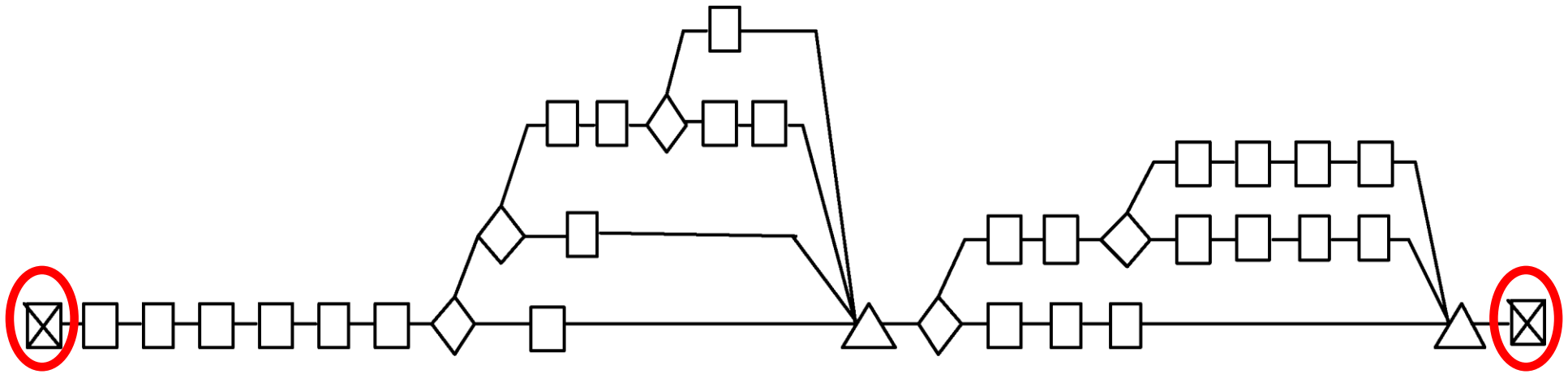
- 由节点和边组成的有向图
  - 节点表示一条或多条语句
  - 边表示节点之间的控制走向，即语句的执行
- 作用
  - 直观反映函数的内部逻辑结构
  - 展示程序中明显的缺陷
  - 揭示程序是否隐含缺陷

1. 孤立节点



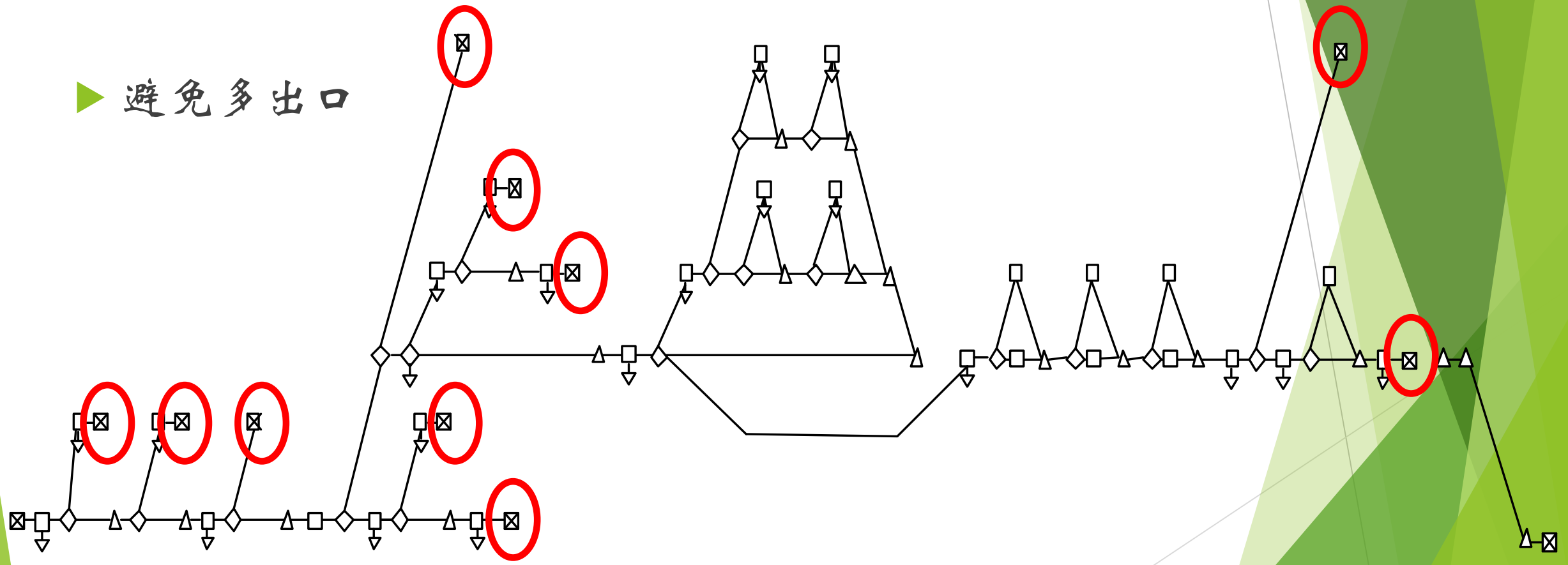
2. 出口节点

- 建议单入单出

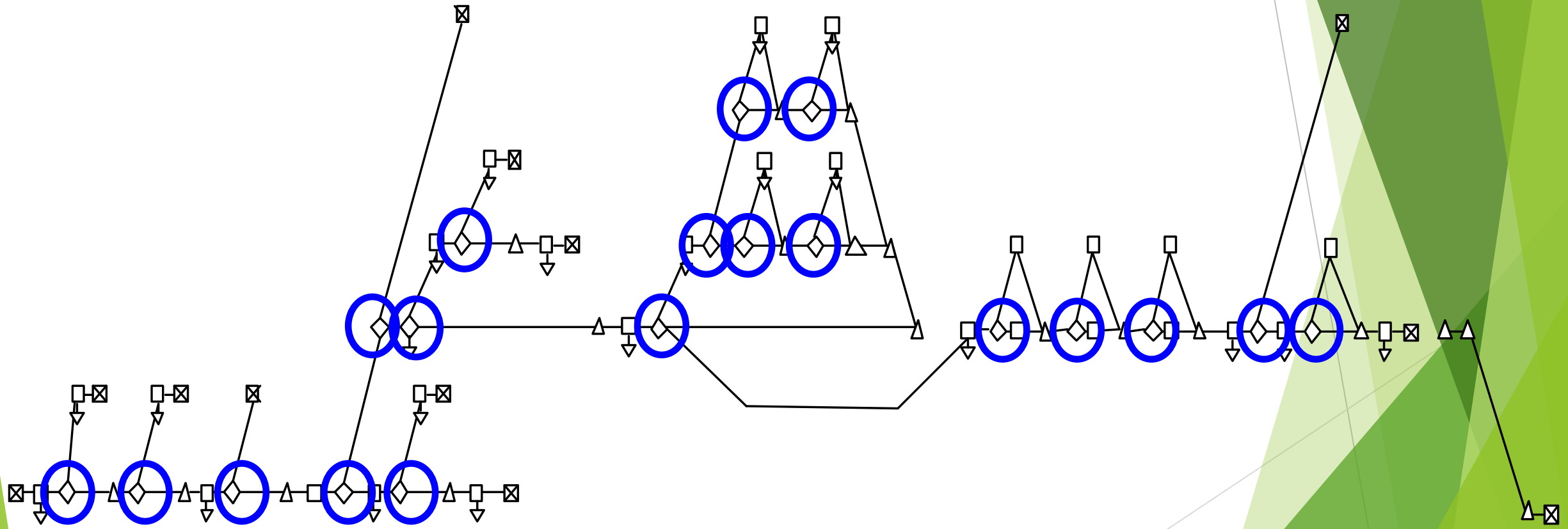


2. 出口节点

- 避免多出口

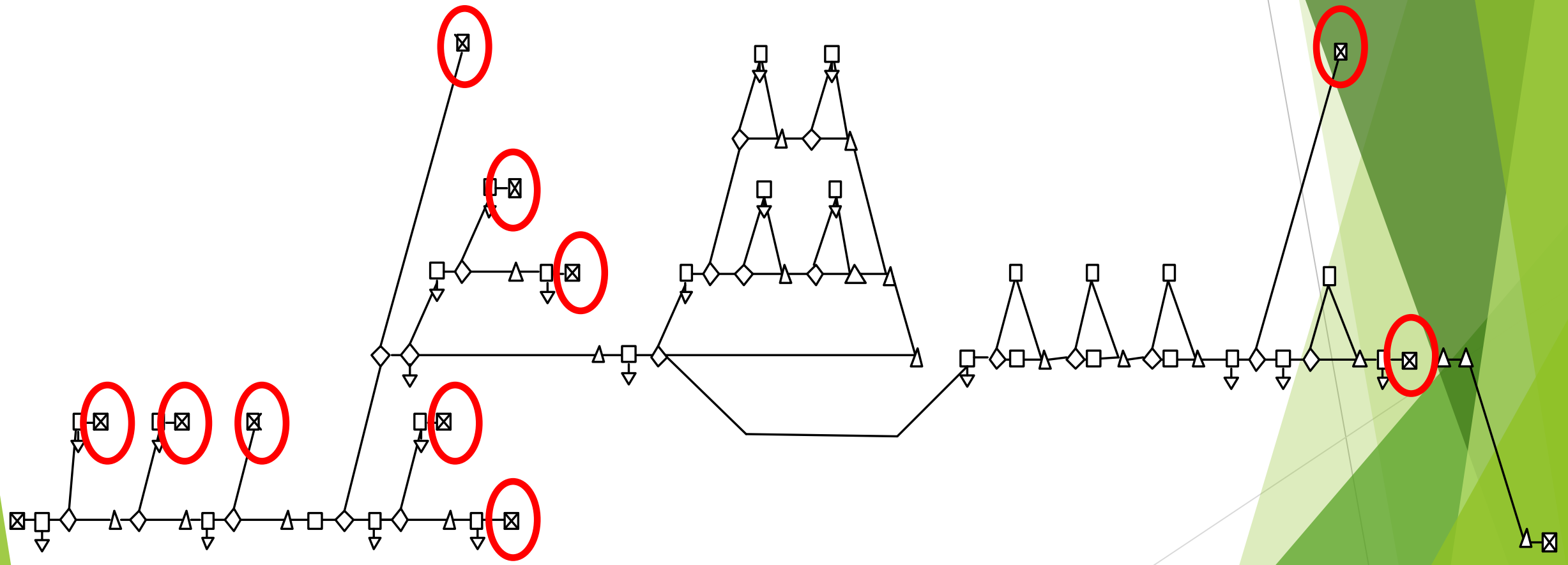


3. 环复杂度



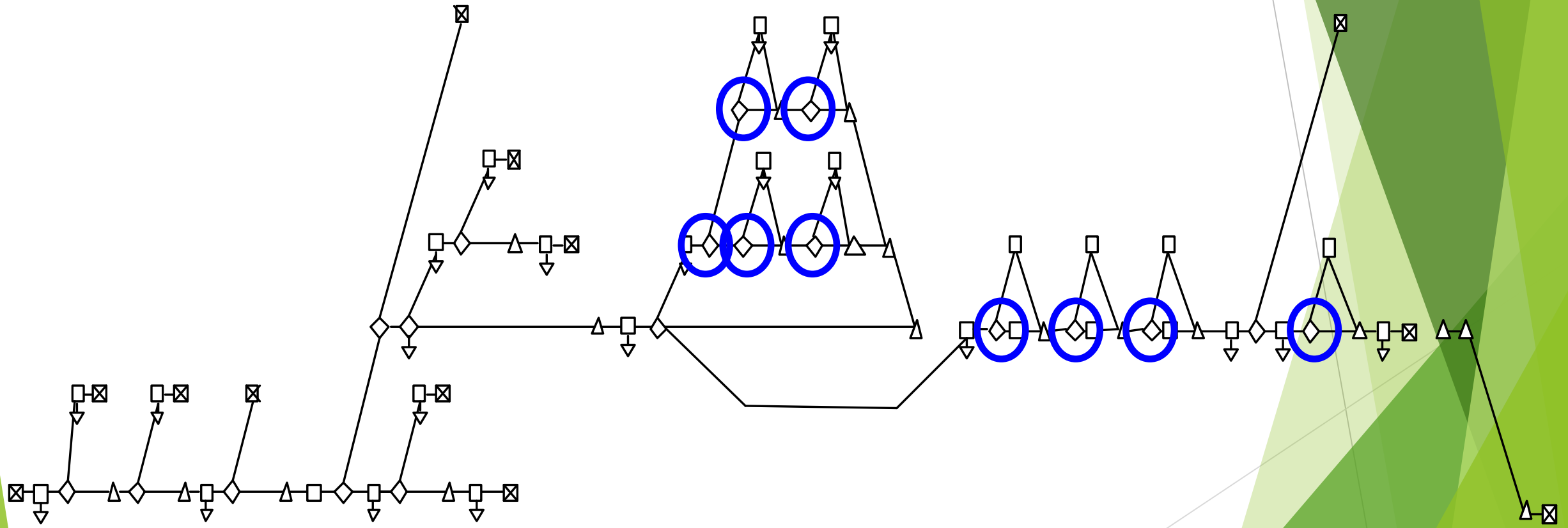
判定节点：19个，环复杂度：20

3. 环复杂度



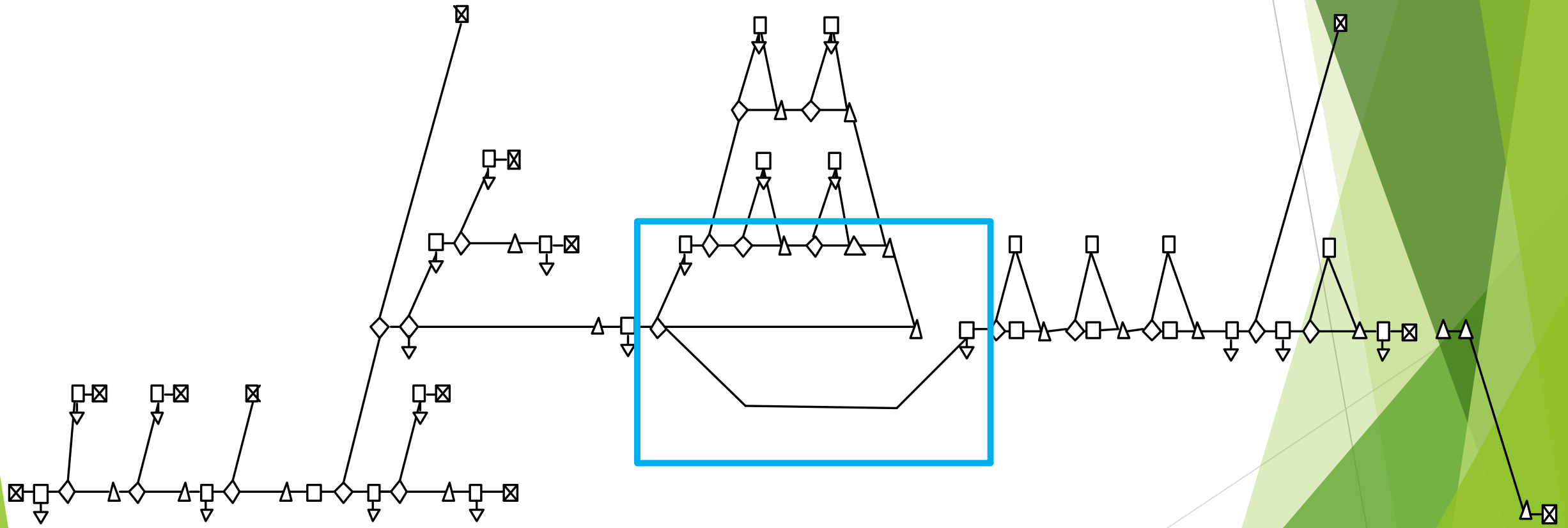
出口节点形成的独立区域

3. 环复杂度



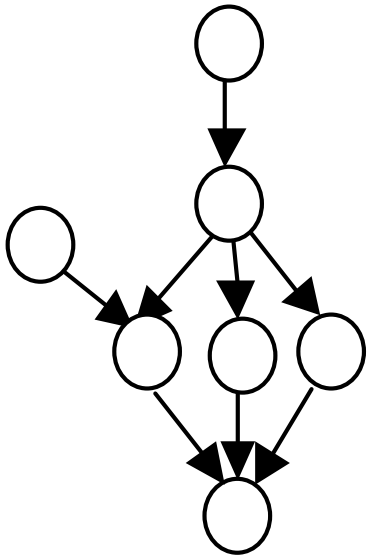
判定结构形成的独立区域

3. 环复杂度

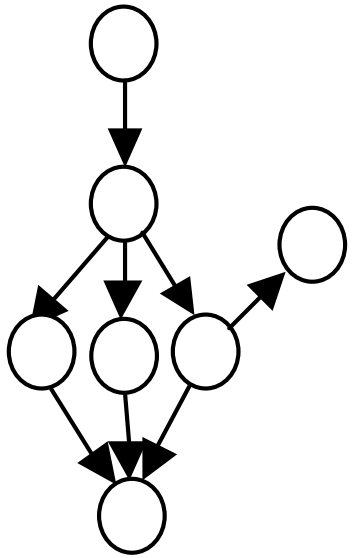


循环结构形成的独立区域

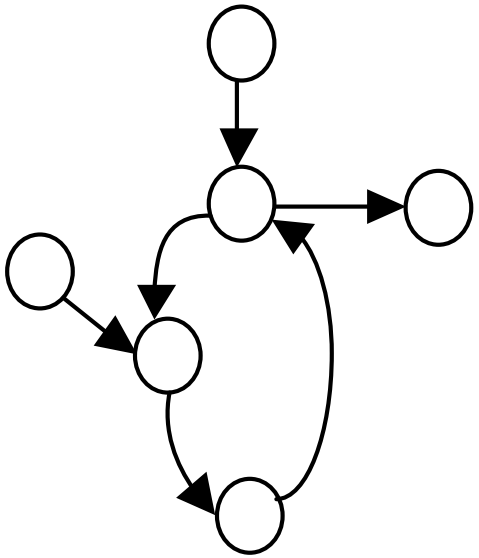
4. 非结构化设计



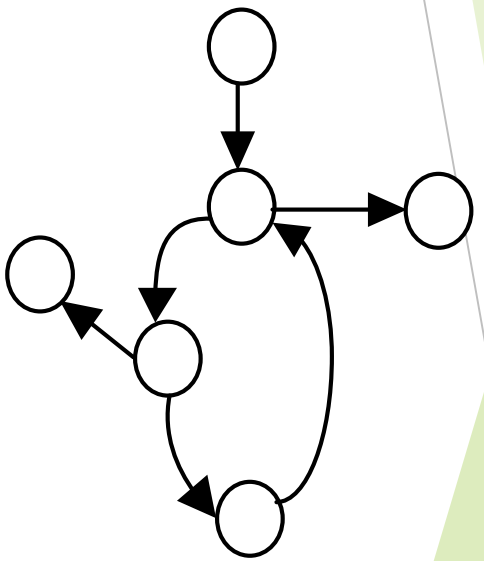
跳入判断



跳出判断

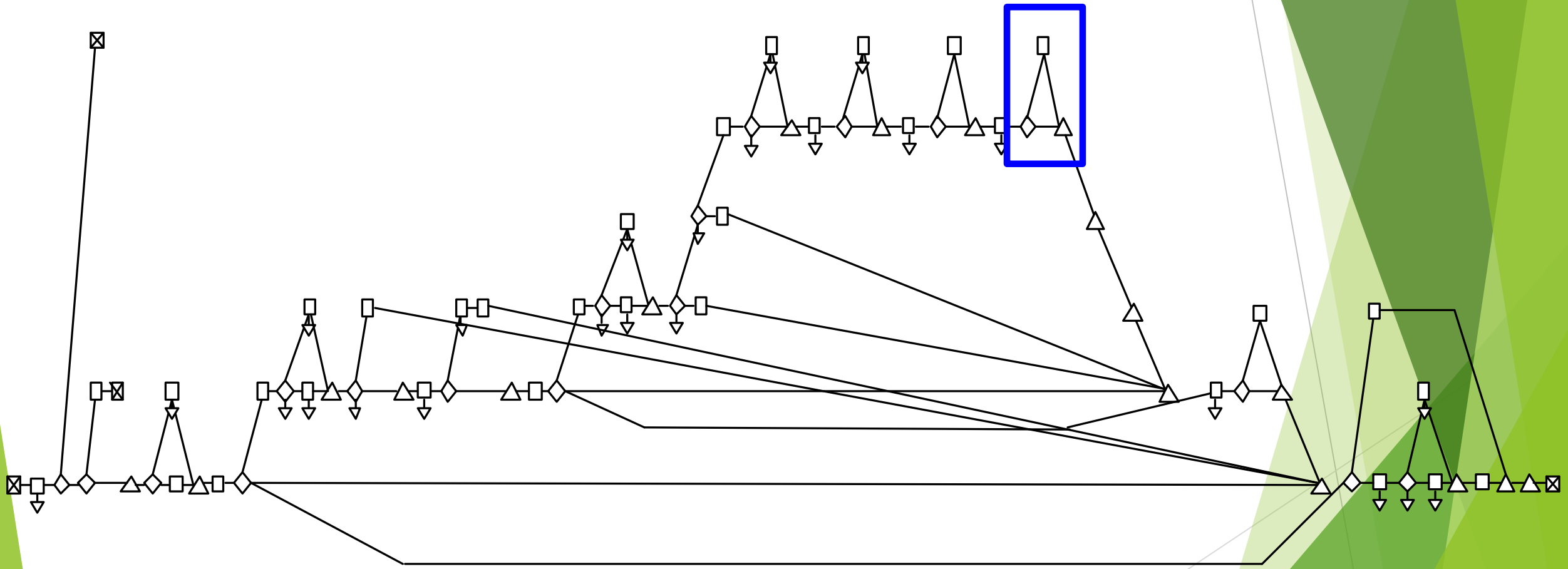


跳入循环

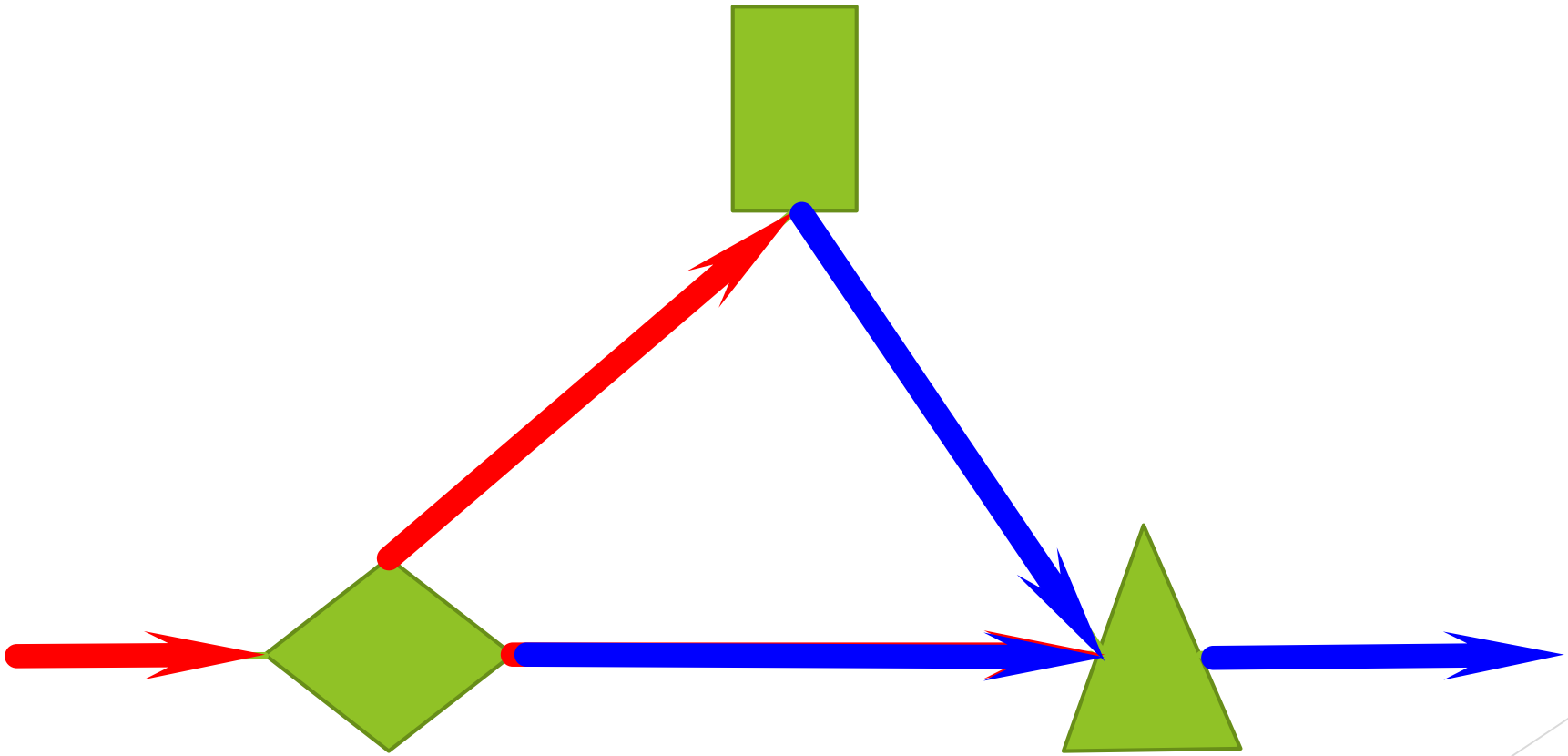


跳出循环

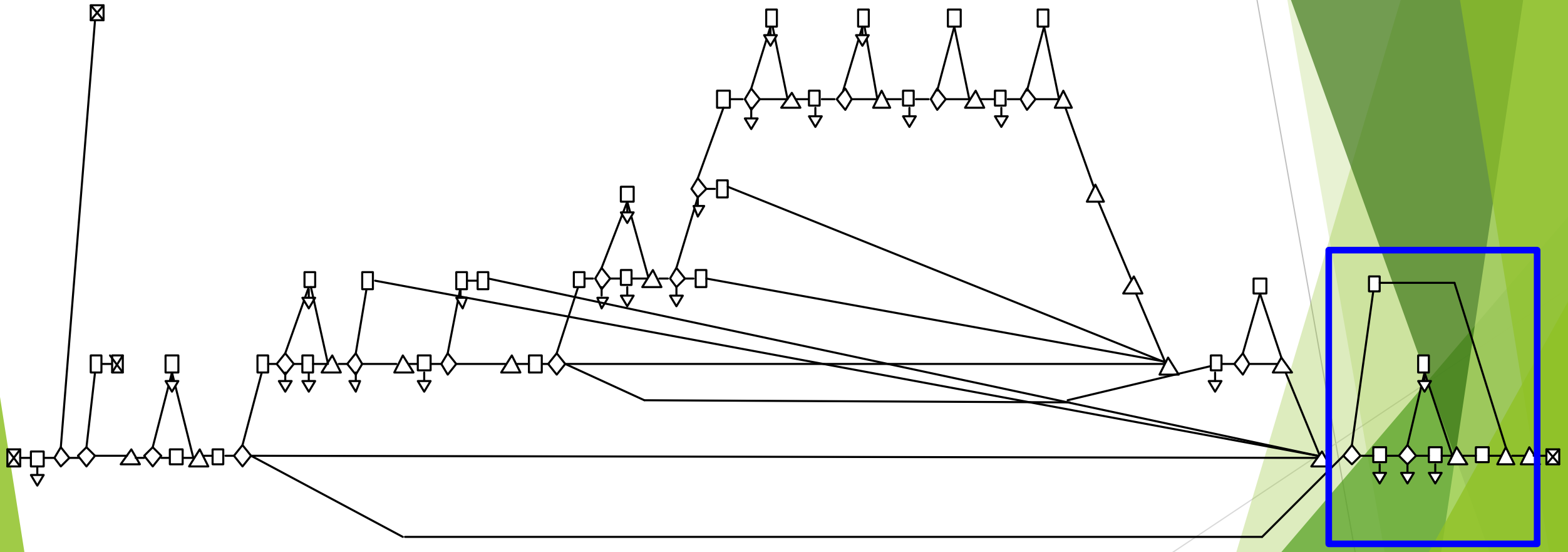
4. 非结构化设计



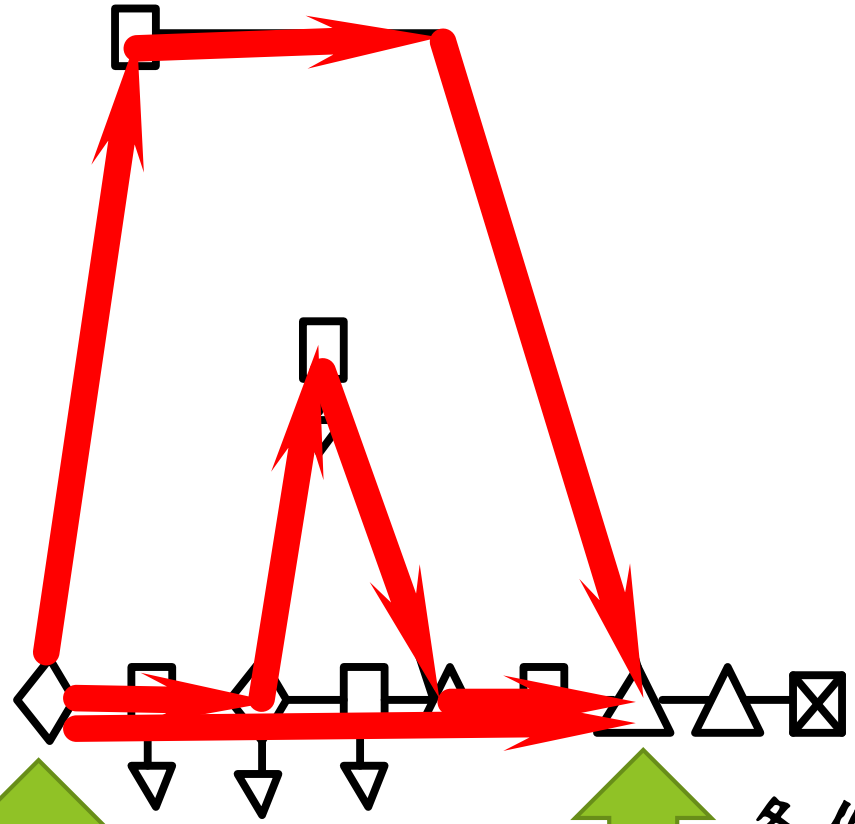
一个典型的结构化设计



4. 非结构化设计



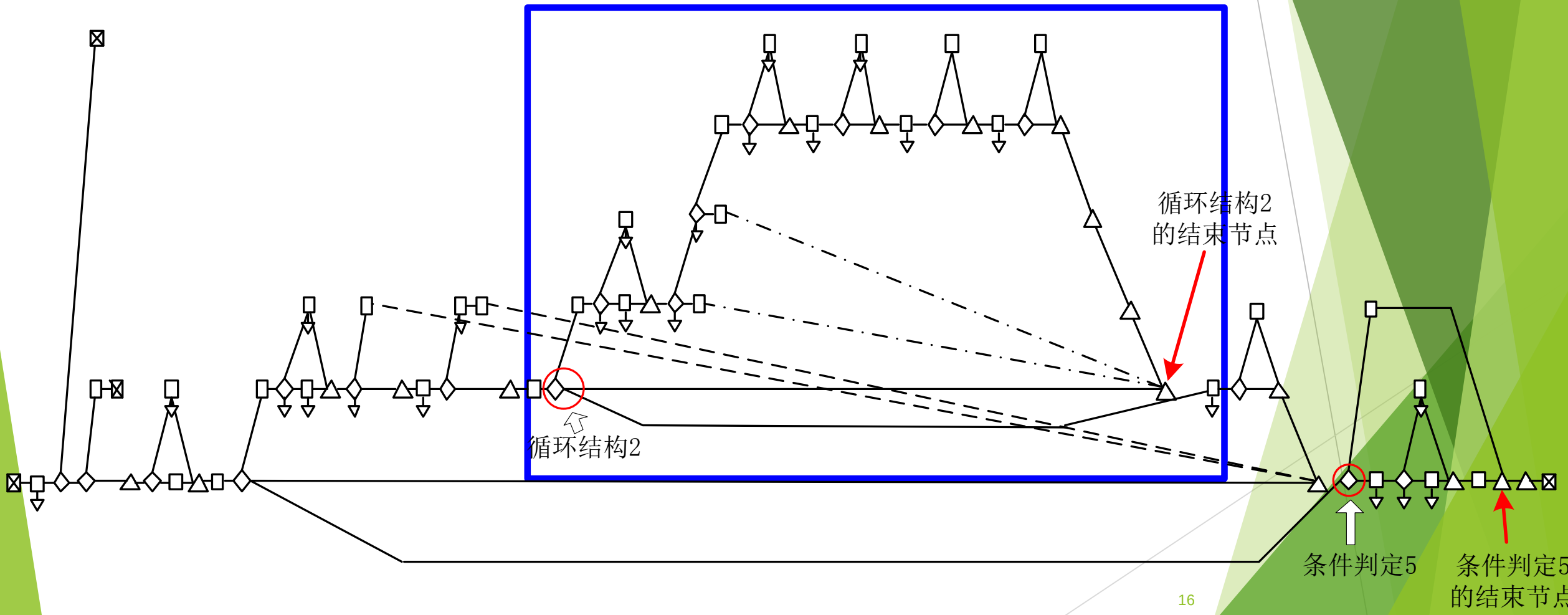
4. 非结构化设计



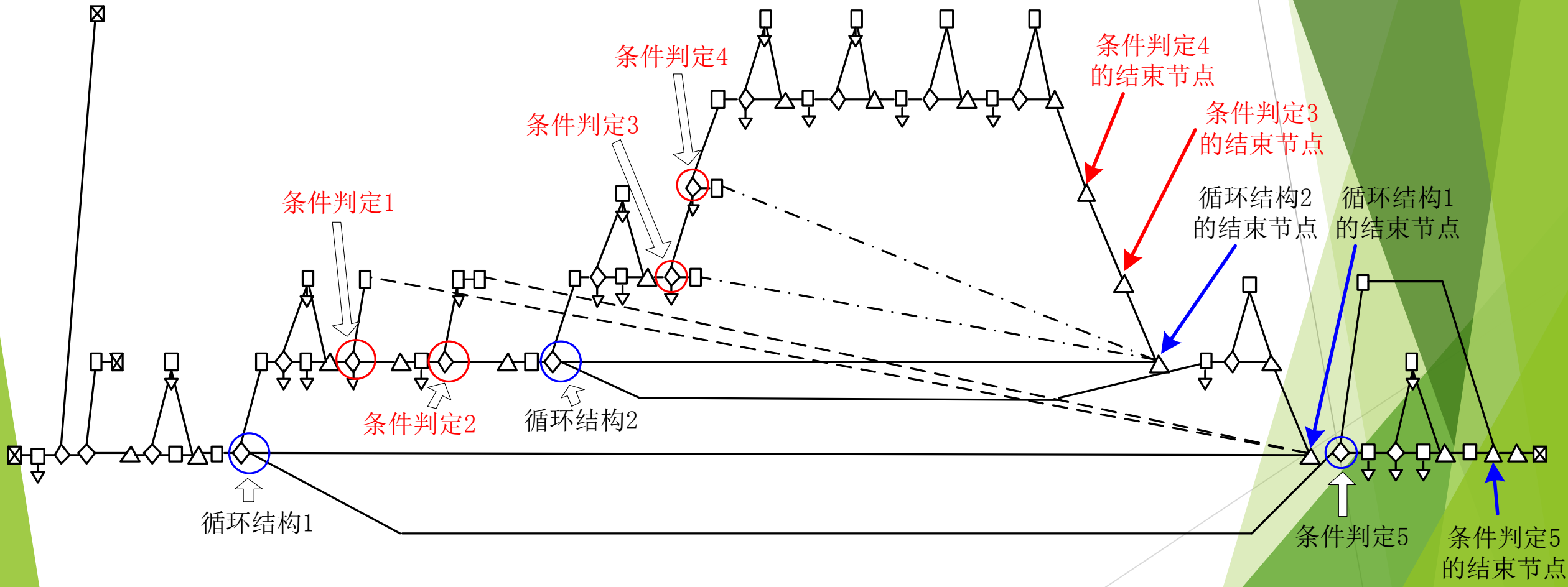
条件判定5
的开始节点

条件判定5
的结束节点

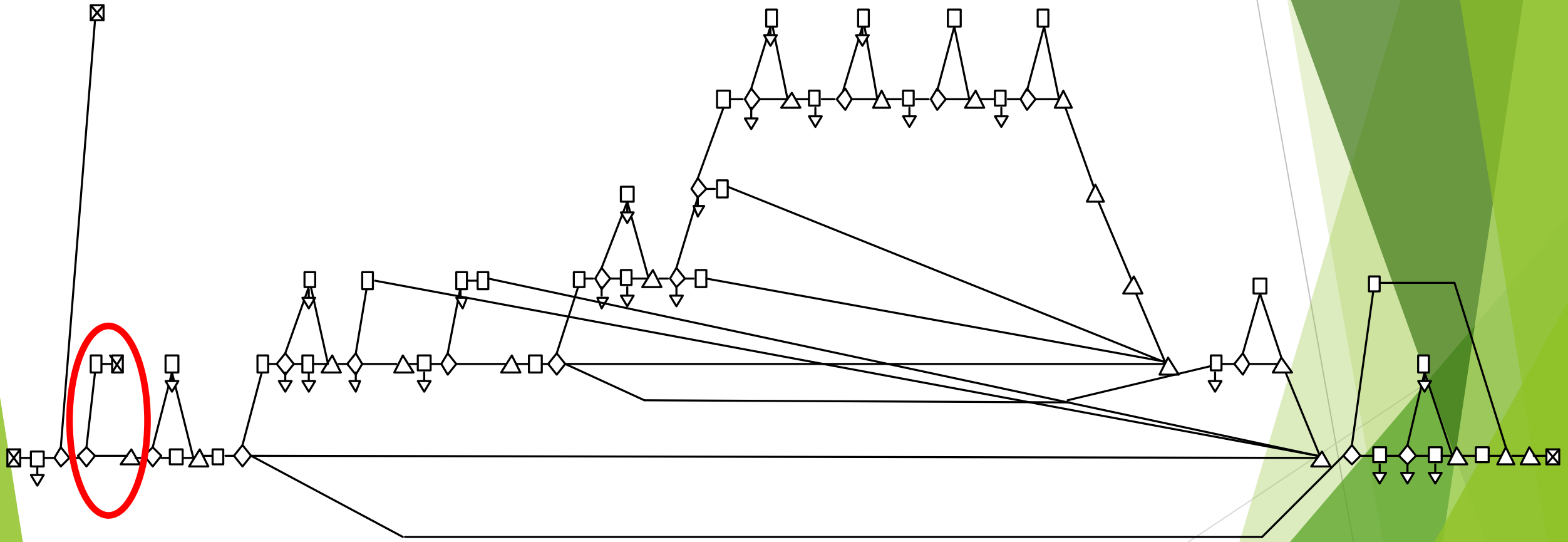
4. 非结构化设计



4. 非结构化设计

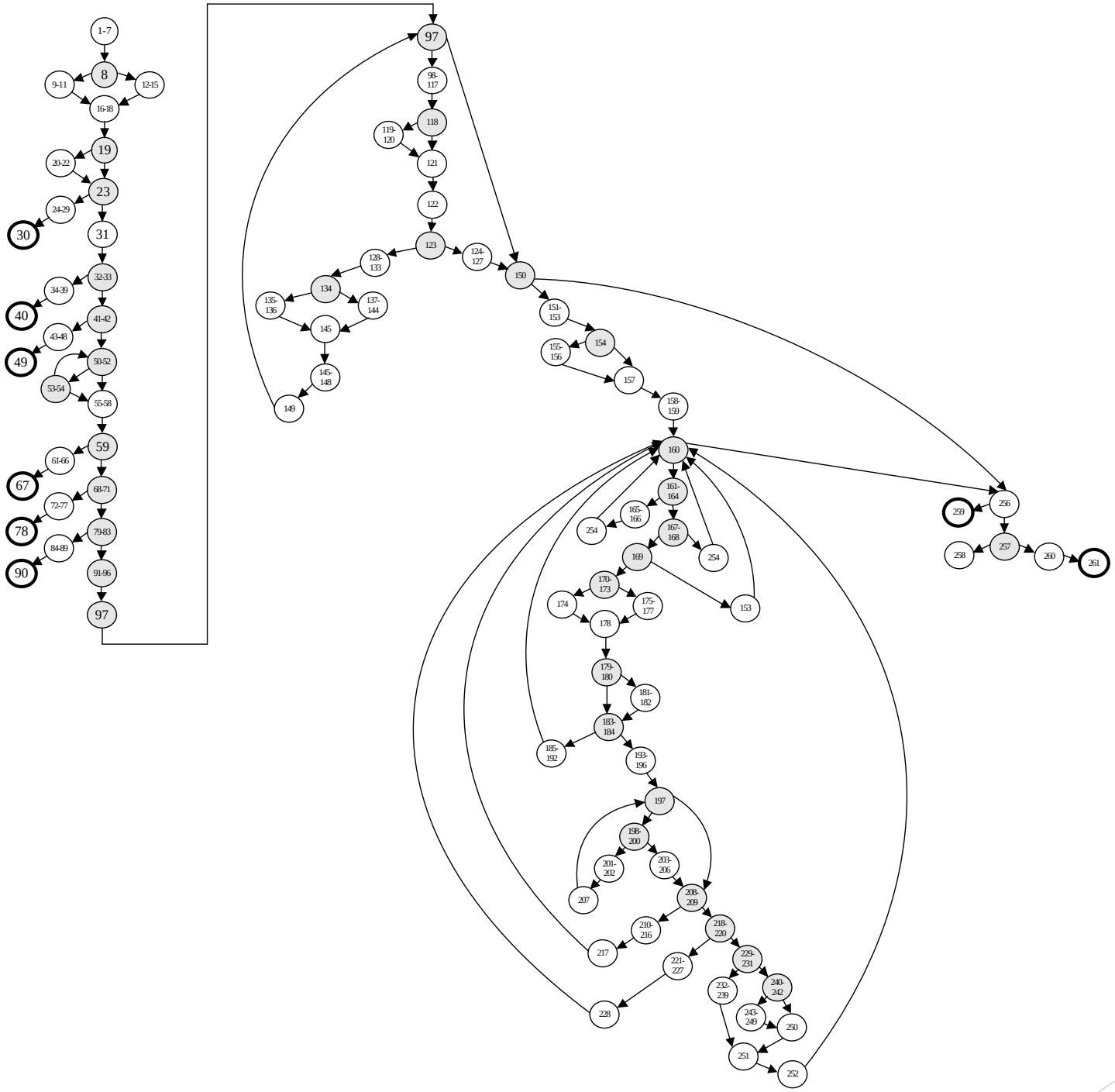

条件判定4
条件判定4
的结束节点
条件判定3
条件判定3
的结束节点
条件判定1
循环结构2
的结束节点
循环结构1
的结束节点
条件判定2
循环结构2
循环结构1
条件判定5
条件判定5
的结束节点

4. 非结构化设计



如何改进程序结构

- 孤立节点：编码时注意避免孤立节点；
- 多出口：尽量避免在同一个函数中多次使用return语句；尽量将有效性校验前置，放到函数外部处理，保证从函数接口传递进来的参数是有效的，避免极短执行路径的函数退出节点；
- 环复杂度：将完成单一功能的语句块改为函数调用的方式；
- 非结构化设计：尽量不使用强制跳转或强制结束语句，如goto、break等语句。


1-7
8
9-11
12-15
16-18
19
20-22
23
24-29
30
31
32-33
34-39
40
41-42
43-48
49
50-52
53-54
55-58
59
61-66
67
68-71
72-77
78
79-83
84-89
90
91-96
97
97
98-117
118
119-120
121
122
123
124-127
128-133
134
135-136
137-144
145
145-148
149
150
151-153
154
155-156
157
158-159
160
161-164
165-166
167-168
169
170-173
174
175-177
178
179-180
181-182
183-184
185-192
193-196
197
198-200
201-202
203-206
207
208-209
210-216
217
218-220
221-227
228
229-231
232-239
240-242
243-249
250
251
252
254
153
256
259
257
258
260
261