### 第二十七次2020年6月22日-6月28日

#### 1 实验部分

其他测试用例进行测试的图像：

地图的截图

描述已自动生成图片包含 游戏机

描述已自动生成

地图的截图

描述已自动生成

将带二元子句对的测试用例转换为带e标记的exactly-one约束测试用例的代码：

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class CNF2Exact {

private void handle(String fileName ,String targetName) throws Exception {

List<String> result = new ArrayList<>();

File target = new File(targetName);

if (!target.isFile()){

target.createNewFile();

}

BufferedWriter writer = new BufferedWriter(new FileWriter(targetName));

FileInputStream tempFile = new FileInputStream(fileName);

InputStreamReader tempInput = new InputStreamReader(tempFile);

BufferedReader tempBuffer = new BufferedReader(tempInput);

String tempString;

String [] tempArray = tempBuffer.readLine().split("\\s");

int varNum = Integer.parseInt(tempArray[2]);

int clauseNum = 0;

while((tempString = tempBuffer.readLine())!=null){

if(tempString.split("\\s+").length == 3){

continue ;

}

clauseNum++;

result.add("e "+tempString + "\n");

}

writer.write("p cnf "+varNum+" "+clauseNum+"\n");

for(String tempResult:result){

writer.write(tempResult);

}

writer.close();

}

public static void main(String[] args) throws Exception {

CNF2Exact test = new CNF2Exact();

for(int i = 1; i<=12 ; i++){

test.handle("C:\\Users\\szeto\\Desktop\\bellSecond\\bell-"+i+".cnf","C:\\Users\\szeto\\Desktop\\bellSecond\\exact"+i+".cnf");

}

}

}

#### 2 学习部分

神经网络编译为可处理的布尔电路。我们考虑布尔电路的易表示性、简洁性和易处理性之间的权衡。布尔电路包含“与”门”、“或”门、逆变器（非门），逆变器仅出现在输入处（电路的输入是变量或它们的取反）。电路的这一子类称为“否定范式（NNF）”电路。 任何带有“与”门，“或”门和逆变器的电路都可以有效地转换为NNF电路。通过在NNF电路的结构上施加属性，可以以简洁性（所得电路的大小）为代价，获得更大的易处理性（在多次时间内执行某些操作的能力）。