【题意简述】

给定一个由与门、或门、异或门、非门组成的无环逻辑电路，定义一个门故障种类如下：

1.总是输出相反的结果

2.总是输出1

3.总是输入0

给定一大包输入和对应输出，判断电路是否正常，若不正常，假设最多一个门故障，输出故障的门与故障类型或无法确定。

【思路分析】

暴力枚举即可。

【复杂度分析】

时间复杂度O(电路输入输出总数\*N^2)

空间复杂度O(N)

【更优方法】

不枚举，直接利用拓扑性计算。

不过没有必要。

【注意点】

从样例可以看出，不一定正常的电路被看成正常电路。

【参考代码】

#include <iostream>

#include <algorithm>

using namespace std;

int n,g,m;

char type[50];

int s1[50],s2[50];

void fetch(int &s){

char a;

int id;

cin>>a>>id;

s=(a=='i')?id-1:id-1+n;

}

void calc(int u,int inp[],int outp[],int err,int etype){

if (outp[u]!=-1) return;

if (u>=n+g){

calc(s1[u],inp,outp,err,etype);

outp[u]=outp[s1[u]];

return;

}

if (u==err && etype==0) outp[u]=0;

else

if (u==err && etype==1) outp[u]=1;

else{

int i1=s1[u],i2=s2[u];

calc(i1,inp,outp,err,etype);

if (type[u]=='n') outp[u]=!outp[i1];

else{

calc(i2,inp,outp,err,etype);

if (type[u]=='a') outp[u]=outp[i1]&outp[i2];

else

if (type[u]=='o') outp[u]=outp[i1]|outp[i2];

else outp[u]=outp[i1]^outp[i2];

}

if (u==err && etype==2) outp[u]^=1;

}

}

void work(int inp[],int outp[],int err,int etype){

for (int i=0;i<n+g+m;i++)

outp[i]=-1;

for (int i=0;i<n;i++)

outp[i]=inp[i];

for (int i=n+g;i<n+g+m;i++)

calc(i,inp,outp,err,etype);

}

int t;

bool fail;

bool ans[50][3];

int main(){

freopen("b.in","r",stdin);

freopen("b.out","w",stdout);

for (int caseI=1;cin>>n>>g>>m && (n || g || m);caseI++){

int i;

for (i=n;i<n+g;i++){

cin>>type[i];

fetch(s1[i]);

if (type[i]!='n')

fetch(s2[i]);

}

for (i=n+g;i<n+g+m;i++){

cin>>s1[i];

s1[i]=s1[i]-1+n;

}

fail=false;

memset(ans,true,sizeof(ans));

cin>>t;

int c\_ans=g\*3;

while (t--){

int inp[50],outp[50],cur[50];

for (i=0;i<n;i++)

cin>>inp[i];

for (i=n+g;i<n+g+m;i++)

cin>>cur[i];

work(inp,outp,0,0);

if (memcmp(outp+n+g,cur+n+g,sizeof(int)\*m)) fail=true;

for (i=n;i<n+g;i++)

for (int j=0;j<3;j++)

if (ans[i][j]){

work(inp,outp,i,j);

if (memcmp(outp+n+g,cur+n+g,sizeof(int)\*m)){

ans[i][j]=false;

c\_ans--;

}

}

}

cout<<"Case "<<caseI<<": ";

if (!fail) cout<<"No faults detected"<<endl;

else{

if (c\_ans!=1) cout<<"Unable to totally classify the failure"<<endl;

else

for (int i=n;i<n+g;i++)

for (int j=0;j<3;j++)

if (ans[i][j]){

cout<<"Gate "<<i-n+1<<" is failing; ";

if (!j) cout<<"output stuck at 0"<<endl;

else

if (j==1) cout<<"output stuck at 1"<<endl;

else cout<<"output inverted"<<endl;

}

}

}

return 0;

}

赵亮