**Term Project - Milestone 2**

309553048\_陳柏丞

* **Execution：**
  + Ubuntu 18.04.5 LTS
    - make
    - ./yeast [input.cnf]
* **Abstract：**
  + Functions：
    - GetData()
    - ReduceClause()
    - UnitPropagate()
    - DPLL()
  + Main：
    - Check input arguments
    - GetData()
    - DPLL()
* **Implementation：**

初始化：將資料讀進一個vector，建立two literal watch 指標

void GetData(

std::string cnf\_file,

int &l\_count,

int &c\_count,

std::vector<std::vector<int>> &box,

std::vector<std::pair<int, int>> &two )

{

std::string s;

std::ifstream cnf(cnf\_file);

if(!cnf) std::cout << "Can't read file: " << cnf\_file << "\n", exit(0);

cnf >> s >> s;

cnf >> l\_count >> c\_count;

int tmp, i=0;

std::vector<int> c\_tmp;

while(cnf >> tmp)

{

if(tmp == 0)

{

box.push\_back(c\_tmp);

c\_tmp.clear();

i++;

continue;

}

c\_tmp.push\_back(tmp);

}

// init iterator

for(auto c:box)

{

if(c.size() > 1) two.push\_back(std::make\_pair(0, 1));

else two.push\_back(std::make\_pair(0, 0));

}

cnf.close();

}

將x=true的clause去除，x=false的clause移除x

bool ReduceClause(

std::vector<std::vector<int>> &box,

std::vector<std::pair<int, int>> &two,

std::vector<int> &ans,

std::queue<int> &q,

int x,

int decision)

{

std::vector<std::vector<int>> box\_tmp;

int i = 0;

for(auto c:box)

{

auto it = std::find(c.begin(), c.end(), x);

if(it == c.end())

{

it = std::find(c.begin(), c.end(), -x);

if(it != c.end())

{

c.erase(it);

if(two[i].first+2 <= box[i].size())

two[i] = std::make\_pair(two[i].second, two[i].first+2);

}

box\_tmp.push\_back(c);

if(c.size() == 1)

{

std::cout << "===> `select: " << c[0] << "\n";

if(ans[abs(c[0])] == -c[0])

{

std::cout << "Conflict on: " << decision << "\n";

ans[decision] = -decision;

for(int i=decision+1; i<ans.size(); i++)

ans[i] = 0;

return false;

}

ans[abs(c[0])] = c[0];

q.push(-c[0]);

}

}

i++;

}

box = box\_tmp;

return true;

}

bool UnitPropagate(

std::vector<std::vector<int>> box,

std::vector<std::pair<int, int>> &two,

std::vector<int> &ans,

std::vector<int> relate,

std::queue<int> &q )

{

std::cout << "prop\_queue: " << q.size() << "\n";

while(!q.empty())

{

int x = q.front();

std::cout << "first literal: " << x << "\n";

q.pop();

for(auto idx:relate)

{

if(box[idx][two[idx].first] == x and two[idx].first+2 <= box[idx].size())

{

std::cout << "first\n";

two[idx] = std::make\_pair(two[idx].second, two[idx].first+2);

}

else if(box[idx][two[idx].second] == x and two[idx].second+1 <= box[idx].size())

{

std::cout << "second\n";

two[idx] = std::make\_pair(two[idx].first, two[idx].second+1);

}

std::vector<int> tmp;

auto it = std::find(box[idx].begin(), box[idx].end(), x);

box[idx].erase(it);

tmp = box[idx];

Print1D(tmp);

}

}

return true;

}

DPLL

bool DPLL(

std::vector<std::vector<int>> box,

std::vector<std::pair<int, int>> &two,

std::vector<int> ans,

std::stack<std::vector<int>> &level,

std::queue<int> &q )

{

int x;

while(!CheckAns(ans))

{

for(int i=1; i<ans.size(); i++)

{

if(ans[i] == 0)

{

x = i;

break;

}

}

std::vector<int> relate;

for(int i=0; i<two.size(); i++)

if(box[i][two[i].first] == -x or box[i][two[i].second] == -x)

relate.push\_back(i);

q.push(-x);

std::cout << "\n===> Select: " << x << "\n";

// update answer in ans

ans[x] = x;

// add ans to level

level.push(ans);

std::vector<std::vector<int>> box\_tmp = box;

for(auto a:ans)

if(a != 0)

if(!ReduceClause(box\_tmp, two, ans, q, a, x)) break;

Print2D(box\_tmp);

// Print

//std::cout << "-----relative clause-----\n";

//Print1D(relate);

//std::cout << "-----iterator-----\n";

//PrintIt(two);

std::cout << "-----Answer-----\n";

Print1D(ans);

while(!UnitPropagate(box, two, ans, relate, q))

{

if(level.size() == 0) return false;

level.pop();

ans[x] = -x;

level.push(ans);

}

}

return true;

}

main

int main(int argc, char\* argv[])

{

if(argc < 2) return 0;

std::string cnf\_file = argv[1];

int l\_count, c\_count;

std::vector<std::vector<int>> c\_box;

std::vector<std::pair<int, int>> two;

GetData(cnf\_file, l\_count, c\_count, c\_box, two);

std::vector<int> ans(l\_count+1 ,0);

std::stack<std::vector<int>> level;

std::queue<int> prop\_q;

level.push(ans);

// Print

std::cout << "-----All clause-----\n";

Print2D(c\_box);

PrintIt(two);

DPLL(c\_box, two, ans, level, prop\_q);

return 0;

}

* **Result：**

這次作業沒有做出來，主要卡在不知道Unit propagate的實作方法，希望可以有補交的機會。