**Homework 2**

309553048\_陳柏丞

* **Execution：**
  + Ubuntu 18.04.5 LTS
    - make
    - ./yeast [input.cnf]
* **Abstract：**
  + Functions：
    - GetData()
    - UpdateTable()
    - ReduceClause()
    - FindUnitClause()
    - DPLL()
    - WriteSatis()
  + Main：
    - Check input arguments
    - GetData()
    - Random a value
    - DPLL()
* **Implementation：**

初始化：將資料讀進一個vector，建立一個table紀錄literal的值

void GetData(

std::string cnf\_file,

int &literal,

int &clauses,

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

std::vector<int> &t )

{

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 >> literal >> clauses;

// no w0

box.push\_back(std::list<int> ());

for(int i=1; i<literal+1; i++)

t.push\_back(i), t.push\_back(-i);

int tmp;

std::list<int> c\_tmp;

while(cnf >> tmp)

{

if(tmp == 0)

{

box.push\_back(c\_tmp);

c\_tmp.clear();

continue;

}

c\_tmp.push\_back(tmp);

}

cnf.close();

}

選了某值x，table就不必再考慮x

std::vector<int> UpdateTable(int x, std::vector<int> table)

{

std::vector<int> t\_tmp;

for(auto ele:table)

if(ele != x && ele != -x) t\_tmp.push\_back(ele);

return t\_tmp;

}

將x=true的clause去除，x=false的clause移除x

std::vector<std::list<int>> ReduceClause(

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

int x )

{

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

box\_tmp.push\_back(std::list<int> ());

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

{

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

if(it == box[i].end())

box[i].remove(-x), box\_tmp.push\_back(box[i]);

}

return box\_tmp;

}

找尋是否為Unit Clause

int FindUnitClause(std::vector<std::list<int>> box)

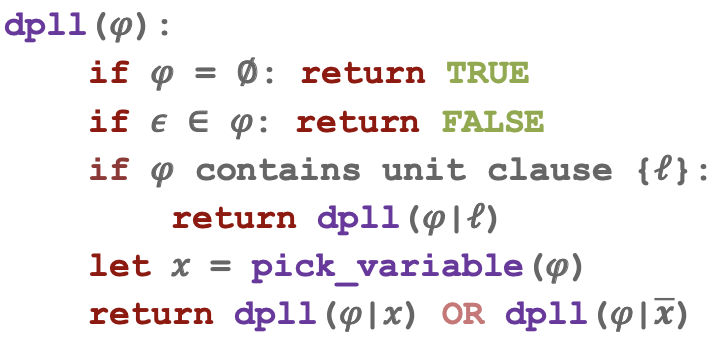
{

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

if(box[i].size() == 1) return box[i].front();

return 0;

}

****

bool DPLL(

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

int x,

std::vector<int> t,

std::vector<int> &ans )

{

std::cout << "\nSelect: " << x;

ans.push\_back(x);

box = ReduceClause(box, x);

if(box.size() == 1) return true;

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

if(box[i].empty()) return false;

if(FindUnitClause(box) != 0)

{

x = FindUnitClause(box);

std::vector<int> t\_tmp = UpdateTable(x, t);

return DPLL(box, x, t\_tmp, ans);

}

x = t[rand() % t.size()];

std::vector<int> t\_tmp = UpdateTable(x, t);

if(DPLL(box, x, t\_tmp, ans)) return true;

else

{

std::cout << "\n\n-----[node] Select branch: " << -x << "-----";

std::vector<int> ans\_tmp;

for(auto ele:ans)

{

if(abs(ele) == abs(-x)) break;

ans\_tmp.push\_back(ele);

}

ans = ans\_tmp;

return DPLL(box, -x, t\_tmp, ans);

}

return false;

}

檢查table的值並寫入out.sat

void WriteSatis(

std::ofstream &output,

std::vector<int> ans,

int literal )

{

output << "s SATISFIABlE\n";

std::vector<bool> ans\_print(literal+1, true);

for(auto ele:ans)

if(ele < 0) ans\_print[abs(ele)] = false;

output << "v ";

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

{

if(ans\_print[i]) output << i << ' ';

else output << -i << ' ';

}

std::cout << "\n\n=> SATISFIABLE\n";

}

初始化資料 → 隨機取值x → if DPLL(x) is true, 寫入out.sat

else DPLL(-x) → if DPLL(-x) is true/false, 寫入out.sat

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

{

if(argc < 2) return 0;

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

int literal, clauses;

std::vector<std::list<int>> clauseBox;

std::vector<int> table, ans;

GetData(cnf\_file, literal, clauses, clauseBox, table);

srand (time(NULL));

int x = table[rand() % table.size()];

std::vector<int> t\_tmp = UpdateTable(x, table);

std::string sat\_file = "out.sat";

std::ofstream output(sat\_file);

if(DPLL(clauseBox, x, t\_tmp, ans)) WriteSatis(output, ans, literal);

else

{

std::cout << "\n\n\*\*\*\*\*[Root] Select branch: " << -x << "\*\*\*\*\*";

if(DPLL(clauseBox, -x, t\_tmp, ans)) WriteSatis(output, ans, literal);

else output << "s UNSATISFIABLE\n", std::cout << "\n\n=> UNSTISFIABLE\n";

}

output.close();

return 0;

}

* **Result：**

1. rand10\_20.cnf



1. rand10\_50.cnf

