Usage:

make
./yasat [input.cnf]

features:

2-literal watching, 1UIP and non-chronological backtracking

Code architecture:

Change the format and renumber the clauses ($x = X * 2$, $\bar{x} = X * 2 + 1$, X from 1 to n)
add $\bar{0}$ to each cluster, which is the level 0 assigned variable
level = 0
while some variable is unassigned :
    add this variable in to a stack
    while stack is not empty :
        v = stack.pop
        assign v
        for all clauses watching v:
            update the watching variables of the clause
            if only one variable is unassigned and others are all false in the clause :
                push this variable into the stack
            if all variable in the clause are false (conflict) :
                if level == 0 :
                    return unsatisfiable
                find 1UIP
                restart_level = the second largest level in 1UIP
                restart from restart_level and add 1UIP into the clauses
    level = level + 1
return the answer

Difficulties:

It takes me a lot of time to think that how to make the code beautiful and efficient. And I still not very satisfy with this code.

Results:

CPU : Intel Core i7-4710HQ @ 2.5 GHz

aim-50-1_6-no-1.cnf
UNSAT
time(sec) 0.00

aim-50-1_6-yes1-1.cnf
SAT
time(sec) 0.00

aim-100-1_6-no-1.cnf
UNSAT
time(sec) 0.00

aim-100-1_6-yes-1.cnf
SAT
time(sec) 0.00

aim-200-1_6-no-1.cnf
UNSAT
time(sec) 0.00

aim-200-1_6-yes1-1.cnf
SAT
time(sec) 0.00

dubois20.cnf
UNSAT
time(sec) 0.00

dubois100.cnf
UNSAT
time(sec) 0.00

ii8a1.cnf
SAT
time(sec) 0.00

ii16a1.cnf
SAT
time(sec) 15.12

ii32a1.cnf
SAT
time(sec) 0.74

jnh1.cnf
SAT
time(sec) 0.02

jnh10.cnf
UNSAT
time(sec) 0.00

jnh11.cnf
UNSAT
time(sec) 0.02

par8-1.cnf
SAT
time(sec) 0.00

par8-1-c.cnf
SAT
time(sec) 0.00

par16-1.cnf
SAT
time(sec) 1.04

par16-1-c.cnf
SAT
time(sec) 5.93

par32-1.cnf
UNKNOWN
over 10 hours

par32-1-c.cnf
UNKNOWN
over 10 hours