Advanced Programming in Artificial Intelligence

Local Search and SAT

Grau en Enginyeria Informàtica
Universitat de Lleida

Josep Argelich

artificial intelligence research group

Local Search

- Introduction
- The Satisfiability Problem
- Random SAT Problem generator
- Local Search Algorithms for SAT

Advanced Programming in Artificial Intelligence

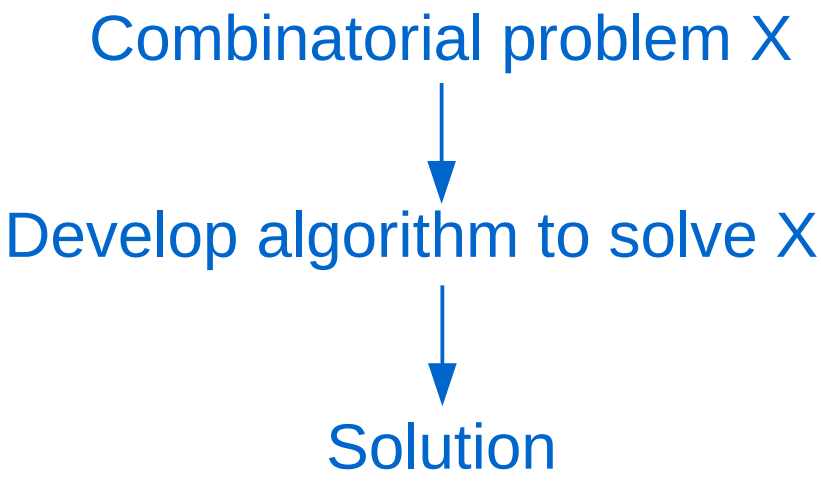
Local Search and SAT

Introduction

artificial intelligence research group

Introduction

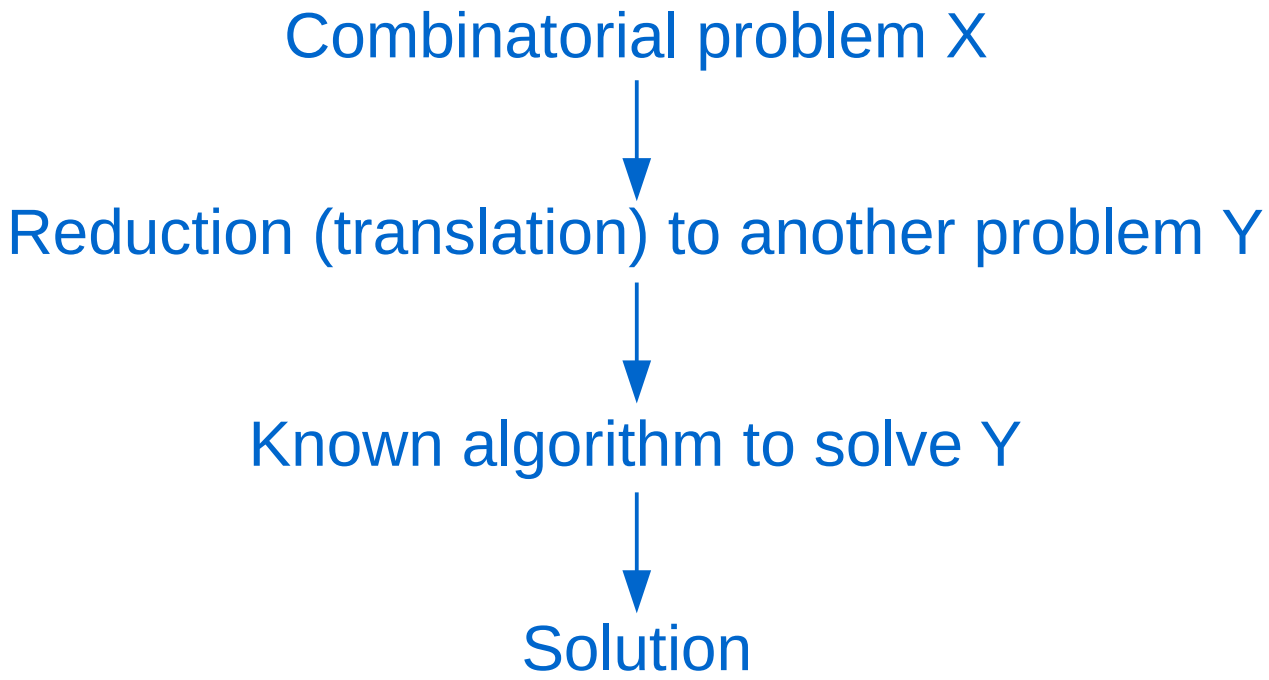
- Real-world problems: Planning, Scheduling, Circuit design, Software verification, Automated theorem proving...



artificial intelligence research group

Introduction

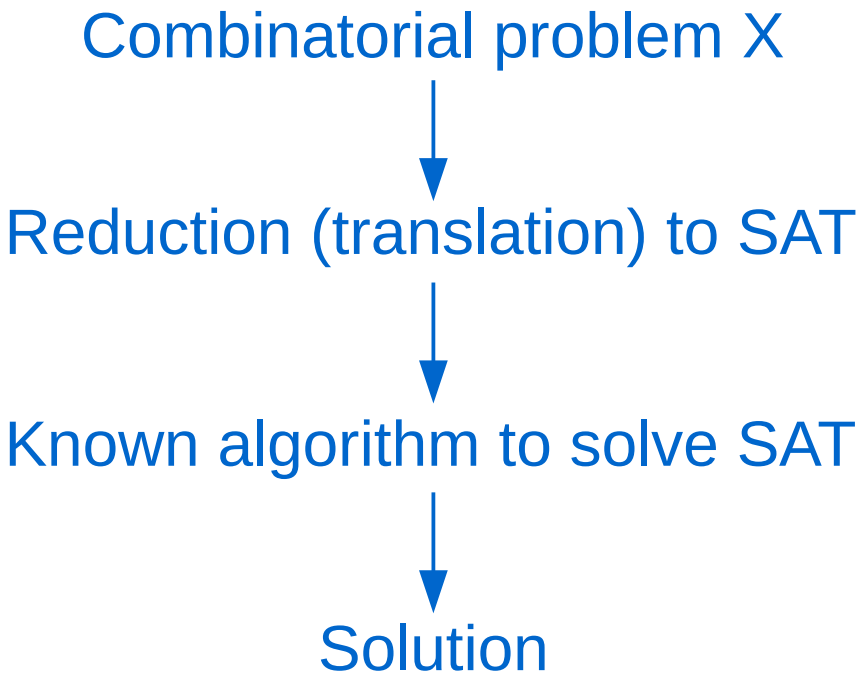
- Real-world problems: Planning, Scheduling, Circuit design, Software verification, Automated theorem proving → SAT



artificial intelligence research group

Introduction

- Real-world problems: Planning, Scheduling, Circuit design, Software verification, Automated theorem proving → SAT



artificial intelligence research group

Advanced Programming in Artificial Intelligence

Local Search and SAT

The Satisfiability Problem

artificial intelligence research group

The Satisfiability Problem

The Boolean Satisfiability problem (SAT)

- SAT is the problem of determining if a propositional formula in Conjunctive Normal Form (CNF) is satisfiable
  - In other words, if there exists an interpretation which satisfies the formula

The CNF Formulas

Set of variables $X = \{x_1, x_2, ..., x_n\}$

- Literal: a positive or negative variable
  - Ex: x_i, $\neg x_i$
- Clause: a disjunction of literals
  - Ex: $x_1 \vee \neg x_2 \vee x_3$, x_2, $\neg x_2 \vee \neg x_3$...
- CNF Formula: a conjunction of clauses
  - Ex: $(x_1 \vee \neg x_2 \vee x_3) \wedge (x_2) \wedge (\neg x_2 \vee \neg x_3)$

  Note: $\vee$ = logical or, $\wedge$ = logical and

The CNF Formulas

- CNF Formula: a conjunction of clauses
  - Ex: $(x_1 \vee \neg x_2 \vee x_3) \wedge (x_2) \wedge (\neg x_2 \vee \neg x_3)$

    Note: v = logical or, ^ = logical and
  - Alternative notation: sets notation
    - Ex: $\{ \{ x_1 \vee \neg x_2 \vee x_3 \}, \{ x_2 \}, \{ \neg x_2 \vee \neg x_3 \} \}$
    - $\square$ denotes an empty clause
    - $\emptyset$ denotes an empty formula

The CNF Formulas

- Interpretation (*I*): is an assignment of truth values {0, 1} (false, true) to variables
- We say that *I* satisfies a:
  - Literal x_i iff $I(x_i) = 1$
  - Literal $\neg x_i$ iff $I(x_i) = 0$
  - Clause *c* iff *I* satisfies at least on literal of *c*
  - CNF Formula *F* iff *I* satisfies all the clauses of *F*
- A CNF Formula *F* is satisfiable iff exist an interpretation that satisfies *F*. Otherwise *F* is unsatisfiable.

The CNF Formulas

- Examples
  - $\{ \square , \{ x_2 \}, \{ \neg x_2 \vee \neg x_3 \} \}$ is unsatisfiable
    - Every formula with an empty clause is trivially unsatisfiable
  - $\{ \varnothing \}$ is satisfiable
    - The empty formula is satisfiable
  - $\{ \{ x_1 \vee \neg x_2 \vee x_3 \}, \{ x_2 \}, \{ \neg x_2 \vee \neg x_3 \} \}$?

The CNF Formulas

- Examples
  - $\{ \square , \{ x_2 \}, \{ \neg x_2 \vee \neg x_3 \} \}$ is unsatisfiable
    - Every formula with an empty clause is trivially unsatisfiable
  - $\{ \varnothing \}$ is satisfiable
    - The empty formula is satisfiable
  - $\{ \{ x_1 \vee \neg x_2 \vee x_3 \}, \{ x_2 \}, \{ \neg x_2 \vee \neg x_3 \} \}$ is satisfiable
    - $x_1 = 1, x_2 = 1, x_3 = 0$ is an interpretation that satisfies the formula
    - There are more interpretations ?

Advanced Programming in Artificial Intelligence

Local Search and SAT

Random SAT Problem generator

artificial intelligence research group

CNF format

- Comments: Line starting with character 'c'
  - Ex: c Random CNF formula
- Problem Line: p cnf *num_variables num_clauses*
  - Ex: p cnf 4 10
- Clauses: List of clauses, one per line
  - Clause: sequence of integers between *num_variables* and *-num_variables* ending with 0 on the same line
    - Positive number → positive literal of the variable
    - Negative number → negative literal of the variable
    - Ex: -3 5 24 -18 0

CNF format

SAT input format example

```
c
c A random CNF formula
c
p cnf 3 4
1 -2 0
-1 2 -3 0
-3 2 0
1 3 0
```

artificial intelligence research group

CNF format

SAT input format example

```
c
c A random CNF formula
c
p cnf 3 4
1 -2 0
-1 2 -3 0
-3 2 0
1 3 0
```

Is there a solution?
1 = true
2 = true
3 = false

Solver output solution:
SATISFIABLE
1 2 -3 0

artificial intelligence research group

Random CNF generator

- Example of a Random k-SAT generator
  - Random generator of CNF problems
  - *K* is the length of the clauses
  - Parameters of the generator
    - Number of variables
    - Number of clauses
    - Length of the clauses (K)
    - Random seed

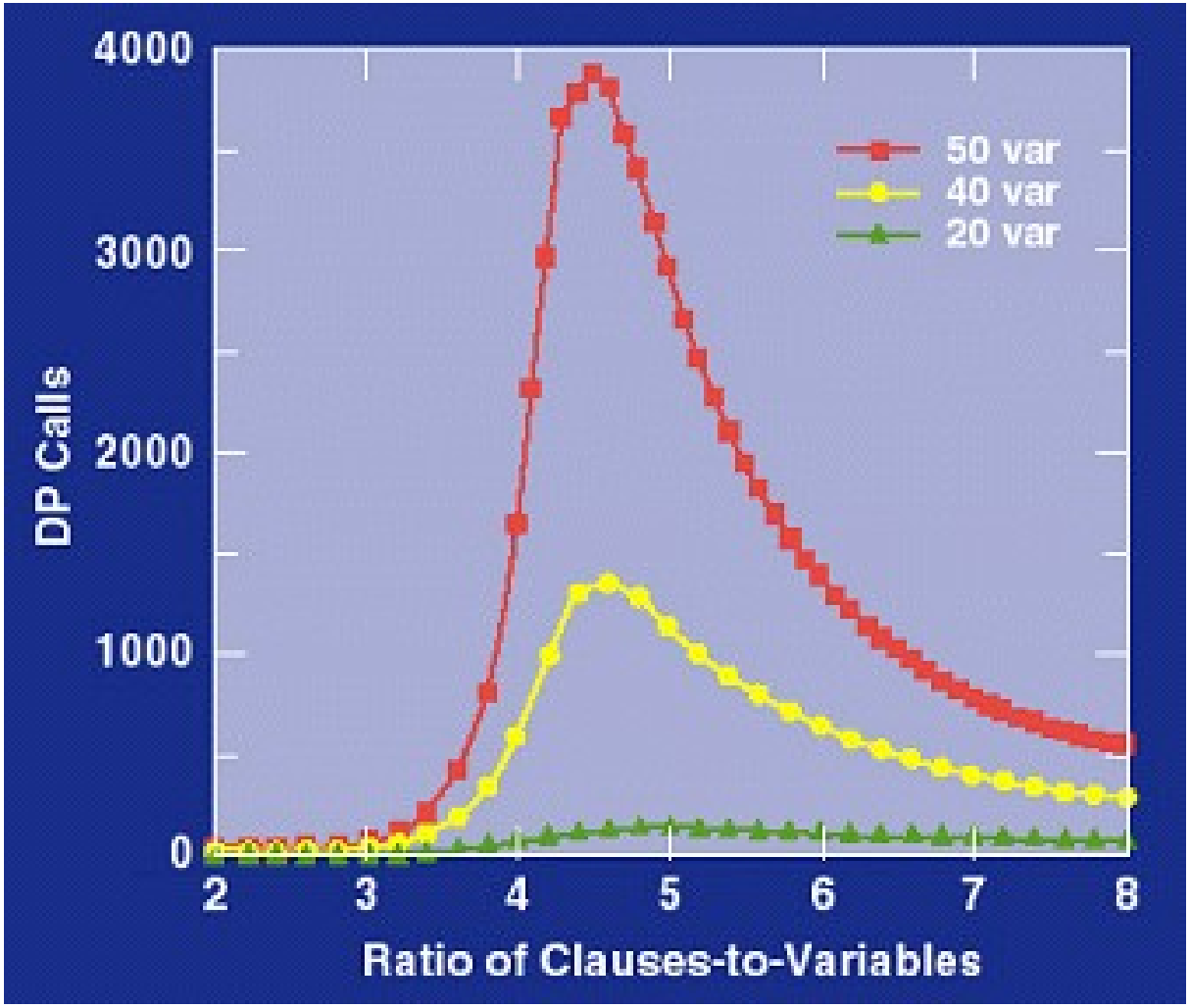
Random 3-SAT Hardness

Easy-hard-easy pattern

The hardness depends on:

$$\frac{\#\text{clauses}}{\#\text{variables}}$$

(peak around 4.25)



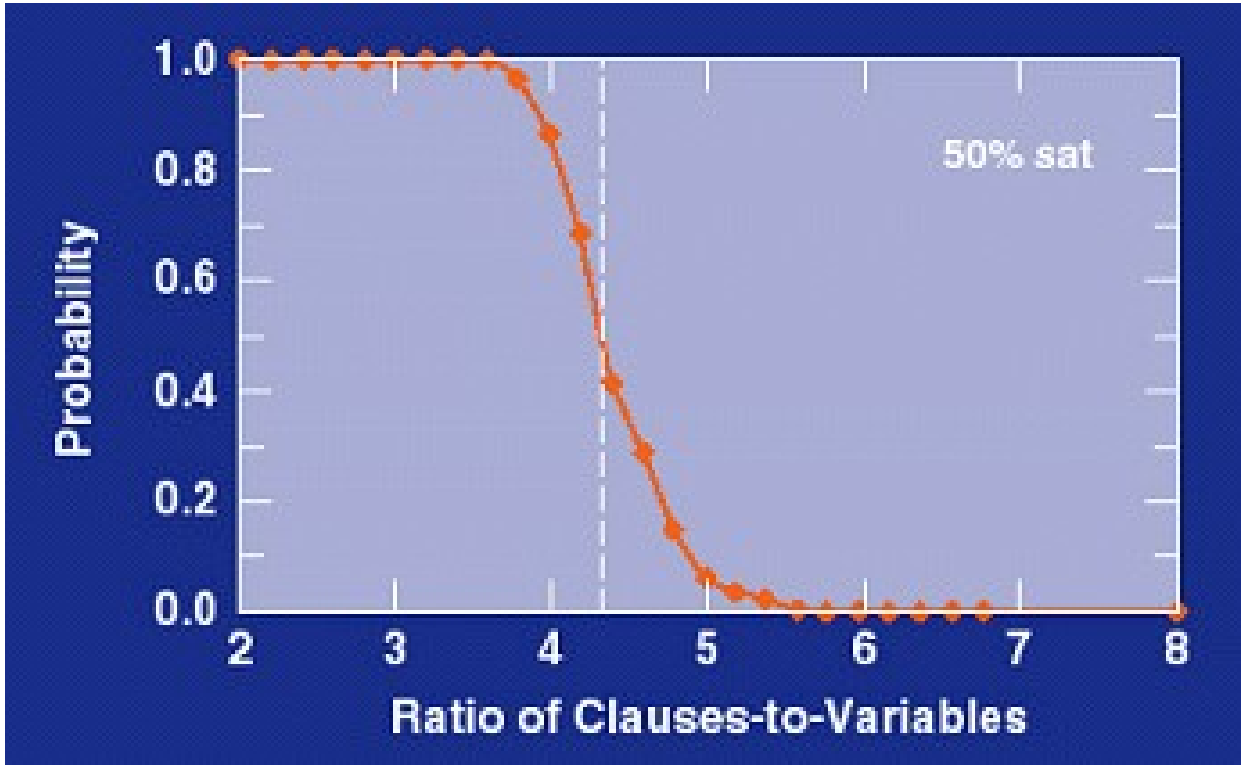
artificial intelligence research group

Random 3-SAT Probability

Phase transition

From satisfiable formulas
to unsatisfiable formulas

(peak around 4.25)



artificial intelligence research group

Experimentation

Reproduce the two previous plots

- Generate the benchmarks (random generator)
- Run the solver with all the benchmarks
  - Send the output to a file (for all the benchmarks)
- Parse the output needed from the results
- Summarize the results
  - Table
  - Plot
  - ...

artificial intelligence research group

Advanced Programming in Artificial Intelligence

Local Search and SAT

Local Search algorithms for SAT

Local Search algorithms for SAT

- Do not follow a systematic method to explore all the search space
  - Not conscious of the explored space
- When they finish...
  - they have found a solution, or
  - they reached a limit of resources
    - In this case, they can not say anything about the solution
- Examples
  - GSAT
  - WalkSat

GSAT

```
F ← input CNF formula
for i = 1 to max_tries do
   I ← random interpretation for F
   for j = 1 to max_flips do
      if I satisfies F then
         return I
      endif
      S ← set of variables s.t. flipping them
         increments more the total number of satisfied
         clauses
      p ← a variable of S
      I ← I with the value of p flipped
   endfor
endfor
return "No solution found"
```

artificial intelligence research group

GSAT

- GSAT can not be used to proof that a formula is unsatisfiable
  - Incomplete procedure
    - If GSAT gets an interpretation, then the formula is SAT
    - If GSAT can not find an interpretation, then we do not know if the formula is SAT or UNSAT
- GSAT is able to solve hard instances with more than 2.000 variables
- Different runs with the same parameters → different solutions
- Problems with local minimums

artificial intelligence research group

GSAT: Escaping from local minimums

- Restarts
  - Restarts the search with a new random interpretation after a given number of flips
- Random walk (GWSAT)
  - with probability ω:
    - Flips a variable from a random unsatisfied clause
  - with probability 1 – ω:
    - Does like GSAT
- Tabu search
  - Some recent flips that take to visited solutions are forbidden
  - GSAT forces to explore new regions in flat areas

Walksat

```
F ← input CNF formula
for i = 1 to max_tries do
    I ← random interpretation for F
    for j = 1 to max_flips do
        if I satisfies F then
            return I
        endif
        C ← a clause of F not satisfied by I
        S ← set of variables that appear in C
        b ← min({broken(p,F,I) | p in S})
        if b > 0 and with probability ω then
            p ← a variable of S
        else
            p ← a variable of S s.t. broken(p,F,I) = b
        endif
        I ← I with the value of p flipped
    endfor
endfor
return “No solution found”
```

artificial intelligence research group

GSAT and Walksat implementations

- There are several implementations of GSAT and Walksat
  - The original: Walksat/GSAT Home Page
    - GSAT is no longer maintained
  - UBCSAT: UBCSAT SATLib
    - Implements a very large sample of solvers
    - Current version 1.1.0

UBCSAT

- ./ubcsat -alg gsat -solve -i test.cnf
  - -alg <name>: name of the algorithm (-ha for the list)
  - -solve: stops when it finds a solution (see -target)
  - -i <file_name>: input CNF instance
  - -runs INT: number of independent tries (default 1)
  - -cutoff INT: number of flips per run (default 100.000)
  - -target INT: target solution quality (default 0)
  - -seed INT: initial random seed
  - -wp <[0,1]>: random walk probability (default 0.5)

Local Search

- Introduction
- The Satisfiability Problem
- Random SAT Problem generator
- Local Search Algorithms for SAT

Bibliography

- “Artificial Intelligence: A Modern Approach”. S. Russell and P. Norvig. Prentice Hall.
- Wikipedia and several Internet sources.

artificial intelligence research group

Advanced Programming in Artificial Intelligence

Local Search and SAT

Grau en Enginyeria Informàtica
Universitat de Lleida

Josep Argelich

artificial intelligence research group