Chapter 27. Other Approaches to Reasoning and Representation

The Quest for Artificial Intelligence, Nilsson, N. J., 2009.

Lecture Notes on Artificial Intelligence, Spring 2012

Summarized by Ha, Jung-Woo and Lee, Beom-Jin

Biointelligence Laboratory
School of Computer Science and Engineering
Seoul National Univertisy

http://bi.snu.ac.kr

Contents

Overview of Chapter 27

- There are some other kinds of approaches to reasoning and representation
- Other kinds of problems
  - Constraints satisfaction problems
  - SAT solver: Propositional logics based solution
  - Representation of text as vectors
  - LSA: a method for text mining

Chapter 27. Other Approaches to Reasoning and Representation

27.1 Solving Constraint Satisfaction Problems

Solving Constraint Satisfaction Problems

- Constraint Satisfaction Problems (assignment problems)
  - Given a set of objects that must be assigned values that satisfy a set of constraints.
  - Constraints: database relation, logical formulas, equations, inequalities, etc.
  - Applications: scheduling, simulation, computer visions, robotics, etc.
  - Ex. Four-Queens problem

Solving Constraint Satisfaction Problems

- AC-3 (A. K. Mackworth)
  - Short for Arc Consistency Algorithm No. 3
  - Basis algorithm of dealing with constraint satisfaction problems
  - Improved to ILOG

Figure 27.4: Alan Mackworth.

- Searching and backtracking
  - When constraint propagation encounters no solution,
  - Ex. c1 = 1
  - Typically, CSP needs searching and backtracking procedures

Chapter 27. Other Approaches to Reasoning and Representation

27.2 Solving Problems Using Propositional Logic

Solving Problems Using Propositional Logic

- No variable logical formulas
  - Ex. Man(Socrates) Mortal(Socrates)
  - Same as propositional logic
  - Too many formulas for cover all of entities
  - Methods for no variable formulas
    - Ex. Who is coming to dinner
      - At least one comes among Ann, Bill, and Charlie
      - If A comes, then B doesn't, if B comes, then C not, and if C comes, then A not
      - $A \vee B \vee C$, $\neg A \vee \neg B$, $\neg B \vee \neg C$, $\neg C \vee \neg A$
- SAT problem
  - The Problem of whether or not there is an assignment of truth values to the variables in a set of clauses such that all of the clauses are satisfied and what those values might be
  - NP-complete
  - Two kinds of solution for SAT problem
    - Systematic methods
    - Local search methods

27.2.1 Systematic Methods

- DPLL
  - Basis of most systematic methods
  - Derived from DP alrorithm by M. Davis and H. Putnam
  - By searching a tree of the possible ways to assign truth values to variables
  - Converting rule
    - 1. In each clause replace the variable just assigned by either a T or an F depending on the branch taken.
    - 2. Eliminate those clauses that contain a T or a ¬F. (These clauses are already satisfied by this assignment.)
    - 3. Eliminate any ¬ T's or F's from any clauses in which they appear. (For the set of clauses to be satisfiable, at least one of the remaining variables in these clauses must have value T.)
    - 4. For any clause that contains just a single variable, set that variable to the value that will satisfy that clause and continue to simplify if possible.

27.2.1 Systematic Methods

- DPLL
  - Termination condition
    - If the set of clauses arrived at is empty, DPLL nishes, having determined that the original set of clauses is satisfiable and that the truth values that have been assigned so far satisfy these clauses.
    - If any of the clauses arrived at along a branch of the tree is empty (that is, there are no more variables left to try to satisfy it), then DPLL has determined that the original set of clauses is unsatisfiable by the truth values that have been assigned so far along that branch. In that case search continues along another branch of the tree if there are still variables with unassigned truth values. If not, DPLL finishes having determined that the original set of clauses is not satisfiable.
  - Example of DPLL

27.2.2 Local Search Methods

- Hill-climbing search
  - A sequence of one-at-a-time modifications to a set of randomly chosen initial truth values for all of the clauses
- GSAT
  - B. Selman, H. Levesque, and D. Mitchell, 1992)
  - Successful on 200,000 variables
  - A kind of hill-climbing search

Figure 27.6: Bart Selman (left) and David G. Mitchell (right). (Photographs courtesy of Bart Selman and of David Mitchell.)

27.2.2 Local Search Methods

- GSAT
  - How it works
    - Start: a random assignment of truth values and evaluates how many clauses this assignment satisfies
    - Flips the truth value of each of the propositions one at a time in turn
    - Termination: satisfies all clauses
    - Local maximum
      - New traverse: restart with different random truth assignment
    - In worst case, spend exponential time due to NP-complete
  - Example: who is coming to party
- WALKSAT (WSAT)
  - An extension of GSAT
  - Instead of the flip with the largest increase, sometimes random flip
- 27.2.3 Application of SAT solvers
  - Generating a plan of action: SATPLAN and Blackbox
  - Verifying programs and digital circuits: BDDs

Chapter 27. Other Approaches to Reasoning and Representation

27.3 Representing Text as Vectors

Representing Text as Vectors

- WWW searching engine
  - Text-based question-answering system
  - Text question → query with logical form
  - Texts and documents are converted to vectors to be compared
- Vector
  - A quantity with direction and magnitude (syn. Point)
  - The number of elements in a vector: the dimension of the vector
  - The similarity of two vectors: the distance between two endpoints

Representing Text as Vectors

- How to convert text to a vector
  - An ordered list of terms (words or phrases) is chosen for the set of documents to be represented by vectors
  - Stemming terms (computing, computed → compute)
  - Eliminating functional words not concerned with contents (if, and, but, ah)
  - Count the occurrence of all terms in the documents
  - Then, a document is represented as follows:
    - Row: occurrence number of the term
    - Column: each term
    - Ex. Why is computer vision important in computer science?

function	computer	calculate	start	display	…	Vision
0	2	0	0	0	…	1

Representing Text as Vectors

- TF-IDF
  - More reasonable vector representation
  - TF: Term frequency
    - Longer documents have more terms
    - The Occurrence of each term / total occurrences for all terms in a document
  - IDF: Inverse document frequency
    - Some terms appears most documents commonly
      - Ex. Be, the, do, etc.
    - Frequently appearing terms in many documents get some penalties
- More vectorizing methods.

Chapter 27. Other Approaches to Reasoning and Representation

27.4 Latent Semantic Analysis

27.4 Latent Semantic Analysis

- LSA
  - Vector representation of text considering order
    - Ex. Dog bites man vs. Man bites dog
  - T. K. Landauer
  - How it works
    - Ex. Small problem
      - Total vocabulary size: 1000 terms
      - Passages → 1000-dimensional vectors
        - divided sections from a document
        - Composed of 100 or so terms
      - 100 passages in the given document

Figure 27.7: Thomas K. Landauer.

27.4 Latent Semantic Analysis

- LSA
  - Subspace: lower-dimension space containin all or most vectors
  - Construct subspace containing 100 vectors with mathematical methods (SVD)
  - Conflation
  - Transformation to lower-dimension: linking terms
    - The reduced-dimension vectors can link together terms from different sections of a text if they occur in passages having a similar meaning even though they never occur in the same passage
  - Measure the similarity between two documents with reduced dimensional vectors
  - LSI (Latent Semantic Indexing): a probabilistic extension

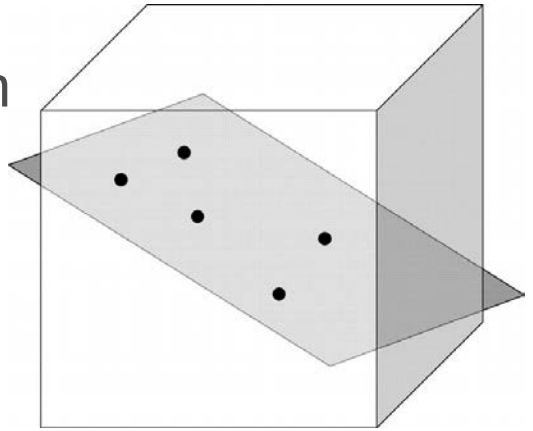

Figure 27.8: A two-dimensional subspace within a three-dimensional space.

Chapter 27. Other Approaches to Reasoning and Representation

Appendix

Solving Constraint Satisfaction Problems

- Four-Queens problem
  - The problem of placing four queens on a 4×4 chessboard in such a way that no queen can capture any other
  - Representation
    - Queen objects: c_1, c_2, c_3, c_4 (subscriptions: column)
    - The value of each queen: row
    - Ex. c2=3 → a queen is in the third row and the second column
  - Constraints
    - Queen's movement rules in the chess game
    - Constraint graph

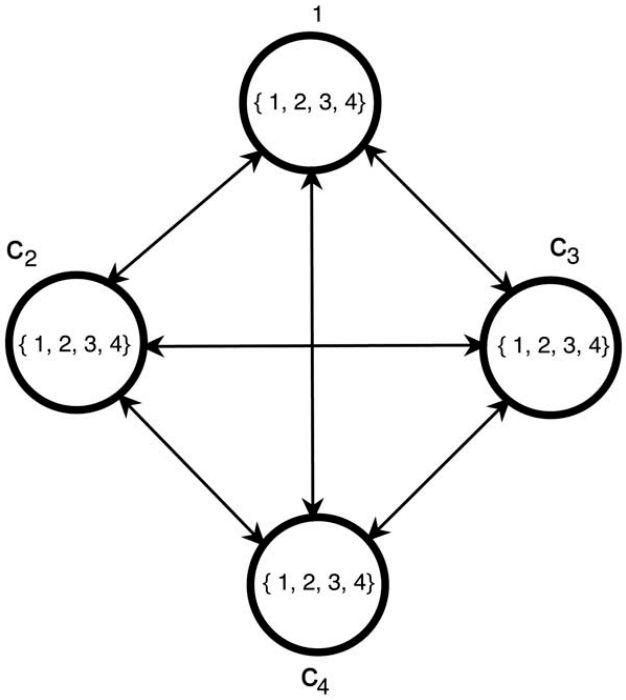

Figure 27.1: A constraint graph for the Four-Queens problem. (From Nils J. Nilsson, Artificial Intelligence: A New Synthesis, p. 185, San Francisco: Morgan Kaufmann Publishers, 1998.)

Solving Constraint Satisfaction Problems

- Four-Queens problem
  - Constraint propagation
    - First, look at the arc from c2 to c1: We can eliminate c2 = 1, c2 = 2, and c2 = 3 because each of those values is inconsistent with the values (there being only one) of c1.
    - Next, look at the arc from c3 to c1: We can eliminate c3 = 2 and c3 = 4.
    - Next, look at the arc from c4 to c1: We can eliminate c4 = 2.

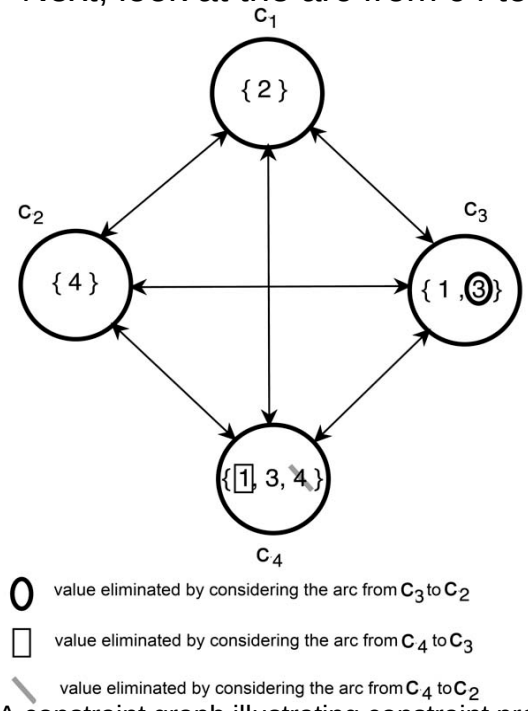

Figure 27.2: A constraint graph illustrating constraint propagation. (From Nils J. Nilsson, Artificial Intelligence: A New Synthesis, p. 187, San Francisco: Morgan Kaufmann Publishers, 1998.)

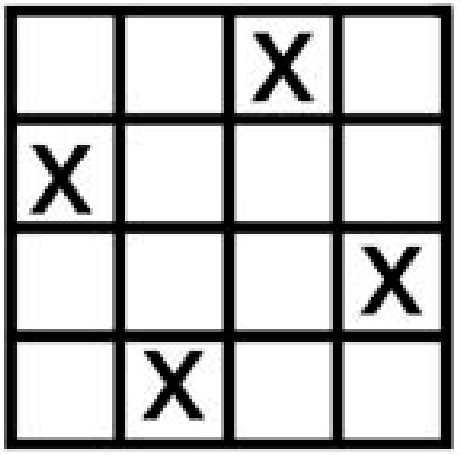

Figure 27.3: A solution to the Four-Queens problem

27.2.1 Systematic Methods

- Ex. Who is coming to dinner
- Termination time depends on searching order
- Depth first search is better
- Some improvements
  - Recursive backtracking search
  - Clause learning method
- zChaff: a web site for SAT

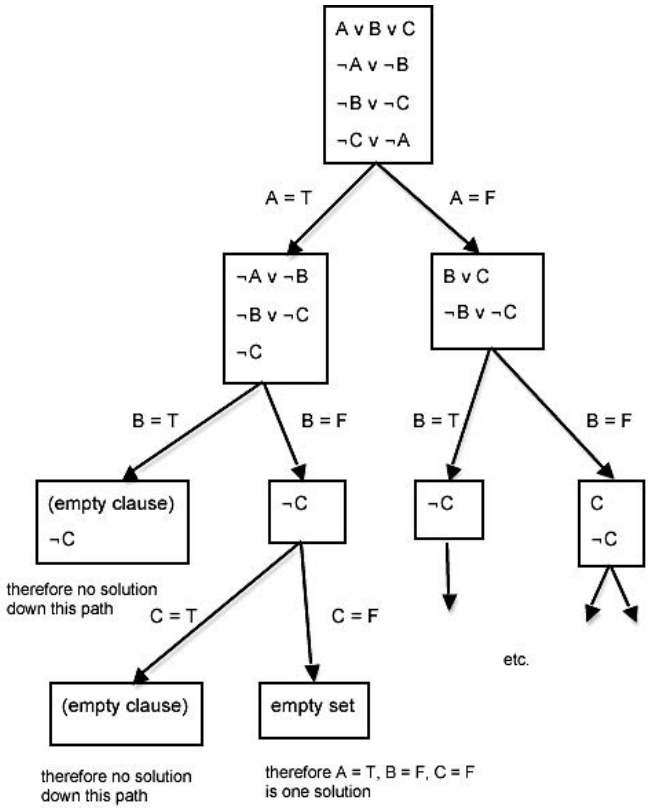

Figure 27.5: A DPLL search tree.

27.2.2 Local Search Methods

- GSAT
  - Ex. Who is coming to party
    - $A \vee B \vee C$, $\neg A \vee \neg B$, $\neg B \vee \neg C$, $\neg C \vee \neg A$
    - Randomly assign values to A, B, and C (T, T, T)
    - Count the truth: 1
    - Flip one value with the largest increase of true clauses(A: T→F)
    - Count the truth: 3
    - ….