AGH University of Science and Technology

Essential Thinking. Introduction to Problem Solving Example Problems III

Antoni Ligęza

Faculty of EEACSE
Department of Automatics

AGH'2011
Kraków, Poland

Outline

References

1. Stuart J. Russel, Peter Norvig: *Artificial Intelligence. A Modern Approach.* Third Edition. Pearson, Prentice Hall, Boston, 2010. `http://aima.cs.berkeley.edu/`.
2. Ivan Bratko: *Prolog Programming for Artificial Intelligence*. Fourth Edition, 2011. Pearson, Addison Wesley, 2012. `http://www.pearsoned.co.uk/HigherEducation/Booksby/Bratko/`
3. Frank van Harmelen, Vladimir Lifschitz, Bruce Porter (Eds.): *Handbook of Knowledge Representation*. Elsevier B.V., Amsterdam, 2008.
4. Michael Negnevitsky: *Artificial Intelligence. A Guide to Intelligent Systems.* Addison-Wesley, Pearson Education Limited, Harlow, England, 2002.
5. Adrian A. Hopgood: *Intelligent Systems for Engineers and Scientists.* CRC Press, Boca Raton, 2001.
6. Joseph C. Giarratano, Gary D. Riley: *Expert Systems. Principles and Programming.* Fourth Edition, Thomson Course Technology, 2005.

References

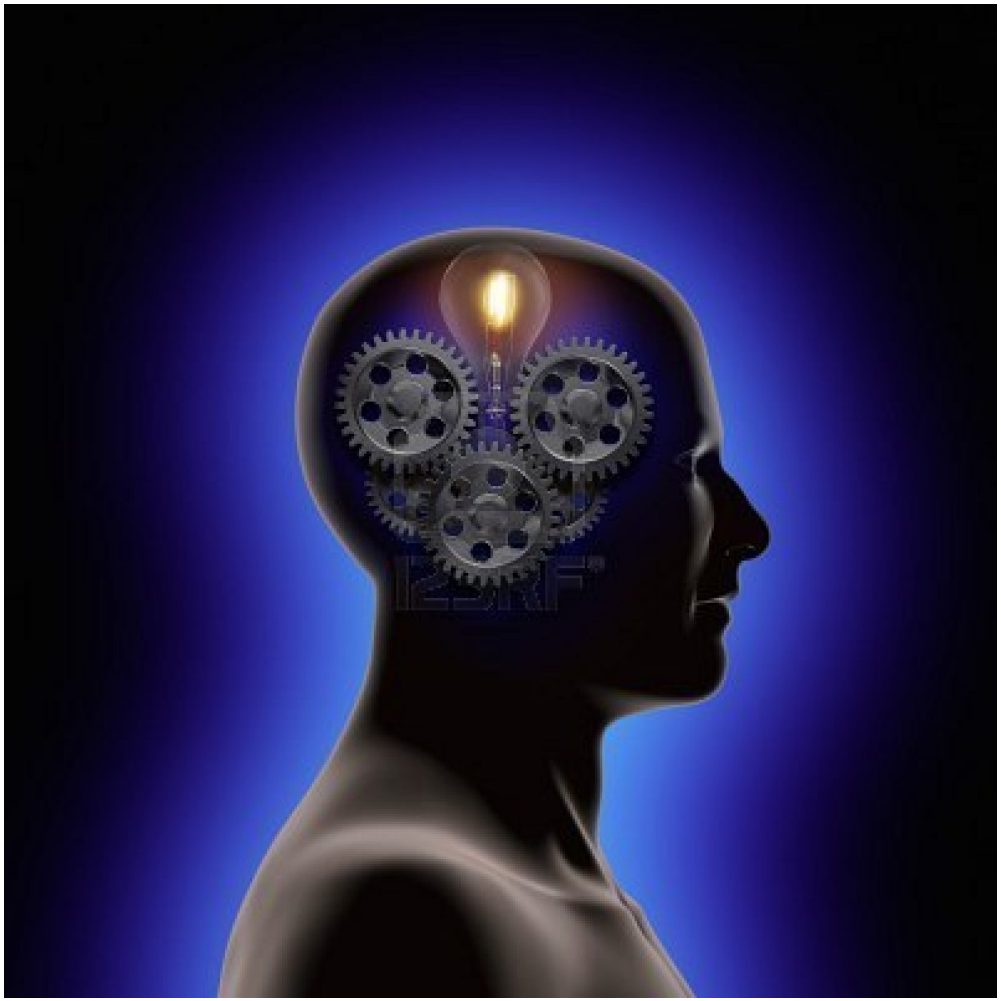
1. George Polya: *How to Solve it?*. Princeton University Press, 1945; PWN 1993. `http://en.wikipedia.org/wiki/How_to_Solve_It`.
2. John Mason, Leone Burton, Kaye Stacey: *Thinking Mathematically*. Addison-Wesley, 1985; WSiP, 2005.
3. Mordechai Ben-Ari: *Mathematical Logic for Computer Science.* Springer-Verlag, London, 2001.
4. Michael R. Genesereth, Nils J. Nilsson: *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc., Los Altos, California, 1987.
5. Zbigniew Huzar: *Elementy logiki dla informatyków*. Oficyna Wyawnicza Politechniki Wrocławskiej, Wrocław, 2007.
6. Peter Jackson: *Introduction to Expert Systems.* Addison-Wesley, Harlow, England, 1999.
7. Antoni Ligęza: *Logical Foundations for Rule-Based Systems.* Springer-Verlag, berlin, 2006.

What is worth learning?

A bit provocative position statement

- Languages — enable communication and knowledge representation;
  Wieviel Sprachen du sprichst, sooftmal bist du Mensch; Goethe
- Problem Solving — analytical thinking; cross-curricular competencies,
- Learning — persistent learning, quick learning, focused learning, learning on-demand, ...

Thinking — What is the Essence of it?



Intelligence: Problem Solving Ability

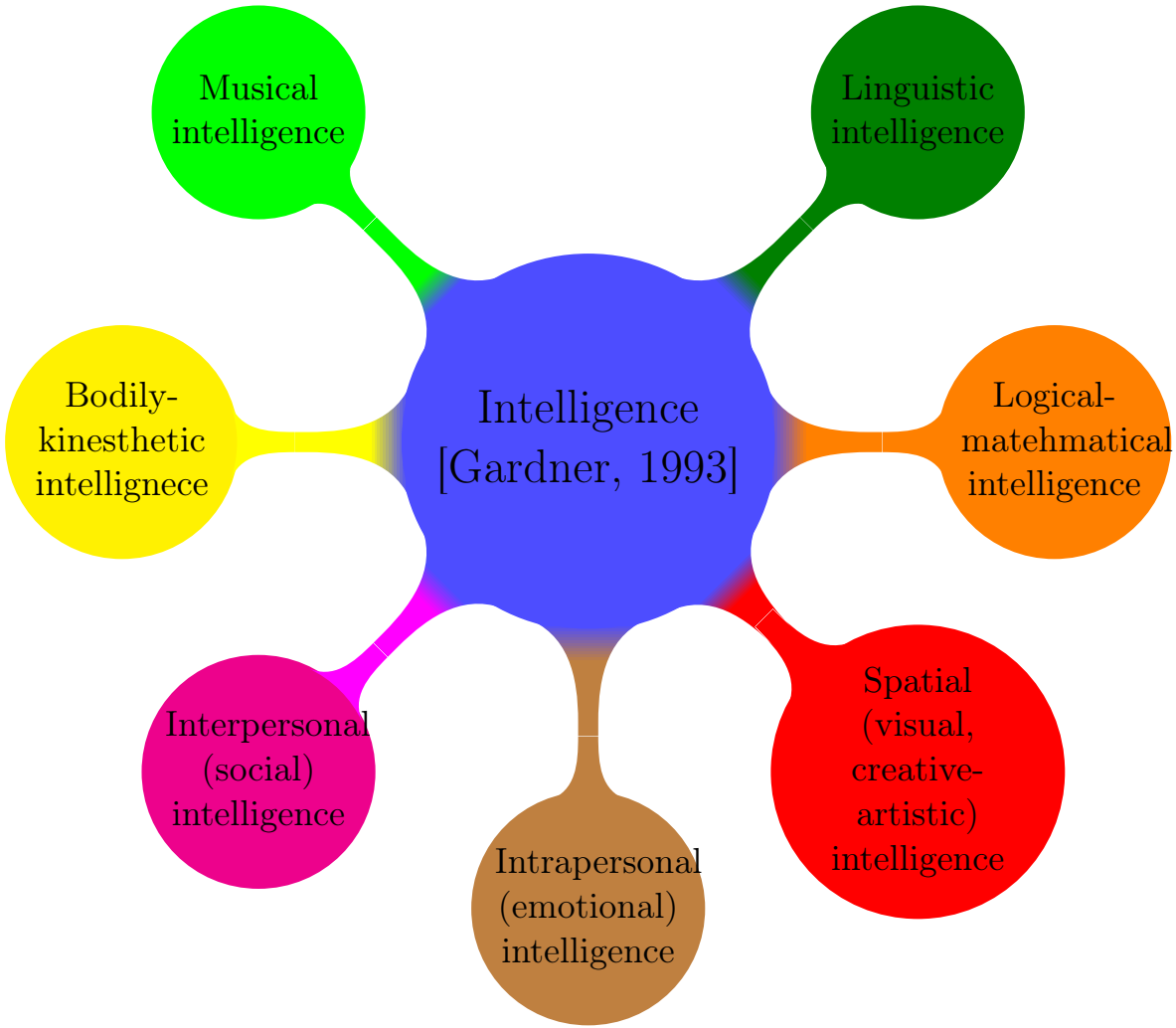
Intelligence: classical understanding

Intelligence — ability to solve new problems.

Intelligence: dimensions

- communication (languages!),
- understanding (models/ontologies),
- reasoning (inference),
- problem solving (search; techniques for PS),
- planning,
- abstract thinking (generalization),
- learning,
- emotional intelligence, freedom, no pure rationality.

Intelligence — What is the Essence of it?



Intelligence: Problem Solving. Questions

Intelligence: questions

- what is intelligence?
- what is the origin of intelligence?
- can intelligence be taught/learnt?
- can we improve intelligence?
- relationship: intelligence, wisdom, knowledge, information, data,...

Intelligence: machines

- can they be intelligent?
- if so, along what dimensions?
- how can man use intelligent machines?
- *can machines be more intelligent than man?*
- *are there any limitations?*

Intelligence: Example Problems

Intelligence: text/speech

- reading and understanding,
- question answering,dialog,
- text abstraction, translation, text synthesis.

Intelligence: vision, images/video

- picture understanding (static),
- video understanding (dynamic),
- similarity analysis, abstraction, synthesis.

Intelligence: motion

- moving toward a goal (static), following the target (dynamic),
- motion planning, obstacle avoidance,
- games (ping-pong, soccer, basket-ball).

General Problem Solver

GPS: Towards General Intelligence

- creation: A. Newell,J.C. Shaw, H.A. Simon; 1957-1959.
- general-purpose problem solver,
- means-ends analysis,
- objects, transformations, differences,
- recursion.

GPS: how it works?

- Method 1: transform object A into object B;
- Methods 2: apply operator Q to A;
- Method 3: reduce the difference d between object A and B.

Means-Ends Analysis

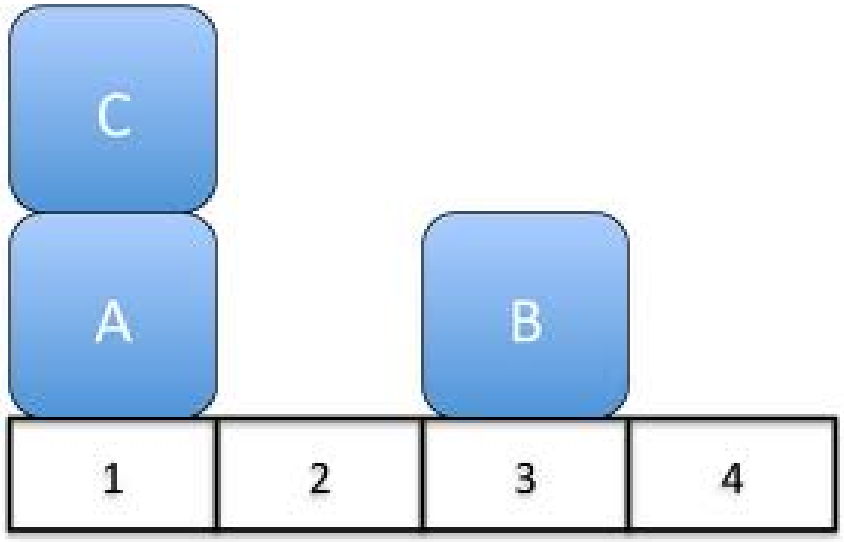
Principles of MEA

- explores the paradigm of goal-based problem solving,
- provides strategy of work at the conceptual level,
- is a universal method,
- roughly based on the conepts of distance and similarity.

MEA: main stages

- compare current state and goal state,
- find differences,
- find operator to reduce the difference,
- apply the operator; produce new state,
- repeat until success;
- backtracking and search are not excluded.

Problems with Means-Ends Analysis



Nonlinear problem

- goal: ON(A,B) and ON(B,C),
- ON(B,C) — one-step, but wrong,
- ON(A,B) — two-steps, but also wrong.

How to Solve It

G. Pólya: Four stages

- understand the problem,
- devise a plan,
- carry out the plan,
- revise/extend.

Auxiliary advice

- if failure, try simpler problem,
- if failure, try related problem,
- partial solutions, auxiliary assumptions, auxiliary/less restrictions.

Understand the problem

Hints and Tips

- What are you asked to find or show?
- Can you restate the problem in your own words?
- Can you think of a picture or a diagram that might help you understand the problem?
- Is there enough information to enable you to find a solution?
- Do you understand all the words used in stating the problem?
- Do you need to ask a question to get the answer?

Devise a Plan

Hints and Tips

- Guess and check,
- Make an orderly list,
- Eliminate possibilities,
- Use symmetry,
- Consider special cases,
- Use direct reasoning,
- Solve an equation.
- Look for a pattern,
- Draw a picture,
- Solve a simpler problem,
- Use a model,
- Work backward,
- Use a formula,
- Be creative,
- Use your head.

A List of Techniques

- Analogy (Mapping to other problem),
- Generalization (Generalization),
- Induction (Induction from examples),
- Variation of the Problem (Modification, change, search),
- Auxiliary Problem (Subproblem, subgoal),
- Here is a problem related to yours and solved before (Pattern recognition, Pattern matching, Reduction; Case-Based Reasoning),
- Specialization (Specialization, instance),
- Decomposing and Recombining (Divide and conquer),
- Working backward (Backward chaining),
- Draw a Figure (Diagrammatic Reasoning),
- Auxiliary Elements (Extension).

Problem Solving: A List of Techniques

- Abstraction: solving the problem in a (simplified) model of the system
- Analogy: using a solution that solved an analogous problem
- Brainstorming: (especially among groups of people) suggesting a large number of solutions or ideas and combining and developing them until an optimum is found
- Divide and conquer: breaking down a complex problem into smaller ones
- Hypothesis testing: assuming a possible explanation to the problem and trying to prove (or, in some contexts, disprove) the assumption
- Lateral thinking: approaching solutions indirectly and creatively
- Means-ends analysis: choosing an action at each step to move closer to the goal
- Method of focal objects: synthesizing seemingly non-matching characteristics of different objects into something new
- Morphological analysis: assessing the output and interactions of an entire system
- Reduction: transforming the problem into another problem
- Research: employing/adapting existing solutions to similar problems
- Root cause analysis: eliminating the cause of the problem
- Trial-and-error: testing possible solutions until the right one is found
- Proof: prove that the problem cannot be solved; fail point = new start

Rational Problem Solving.

A Systems Science Approach

- Observation, analysis — understanding,
- Problem detection, problem statement,
- Goal definition,
- Performance indicators,
- Constraints definition,
- Model development:
  - inputs, outputs, noise, observations,
  - task definition,
  - determining transfer function,
- Problem solution,
- solution analysis, verification, optimization,
- lessons learned, repeat cycle, spiral model, generalization.

What do we need to solve a problem?

9 components

- Knowledge representation,
- Inference rules (legal mooves),
- Inference control (avoid combinatorial explosion),
- Knowledge acqusition,
- User interface (picture!),
- Verification and Validation,
- Modification, extention, adaptataion, learning,
- Abstraction, generalization,
- Automated approach.

Knowledge Representation Techniques

KR

- numeric (numbers, vectors, matrices, functions, equations),
- qualitative (intervals, symbolic, $\{-, 0, +\}$,
- algebraic (sets, relations, structures),
- logical formalisms (facts, formulas, rules),
- rule-based systems, rules,
- graphs, semantic networks,
- frames, structural (objects),
- pictures (diagrams, schemes, blocks),
- combined (e.g. XTT).

Inference Techniques

Patterns of inference

- Abstraction (generalization),
- Specjalization,
- Pattern Matching; Case-Based Reasoning, analogy,
- Logical inference (deduction, abduction, induction),
- Rule-Based Inference (forward, backward, top-down),
- Search algorithms,
- Problem reduction (AND-OR graph search),
- Constraint Satisfaction Techniques,
- Consistency-Based Reasoning,
- Graph Transformations, Graph Grammars,
- Numerical Procedures (e.g. optimization),

Inference Control Techniques

Efficient search for solution

- search-space selection,
- systematic, blind search,
- heuristic search,
- search with constraints,
- problem reduction,
- constraint relaxation,
- mini-max strategies,
- elimination of cases (tabu search).

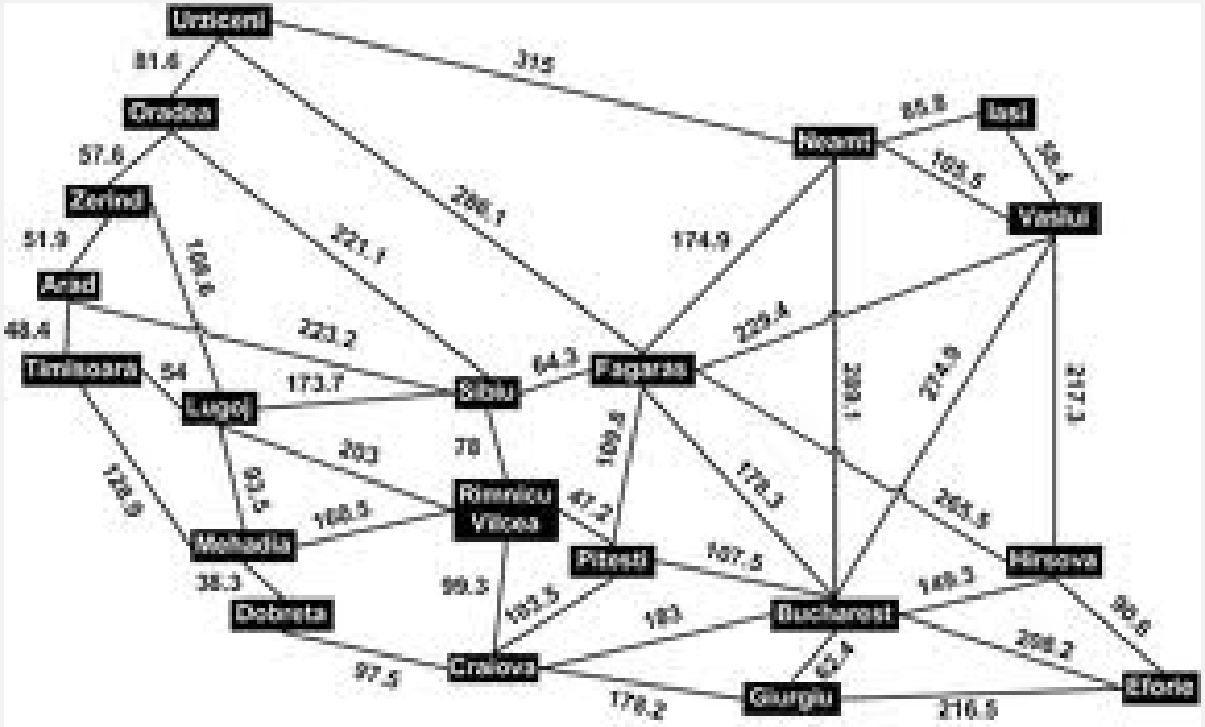
Types of Problems

An informal classification

- FORWARD CHAINING (deduction, rules, patterns),
- BACKWARD CHAINING (abduction, diagnostics, hypothetical reasoning),
- UPWARD INFERENCE (induction, model building),
- SEARCH — graph search, path finding,
- PLAN — plan generation,
- REDUCT — AND-OR graph search, AND-OR plans,
- GAME - adverserial search,
- CSP, CLP — search with constraints,
- OPT — optimal solution search,
- CI — Computational Intelligence problem (NN, Fuzzy, k-NN).

Three generic examples

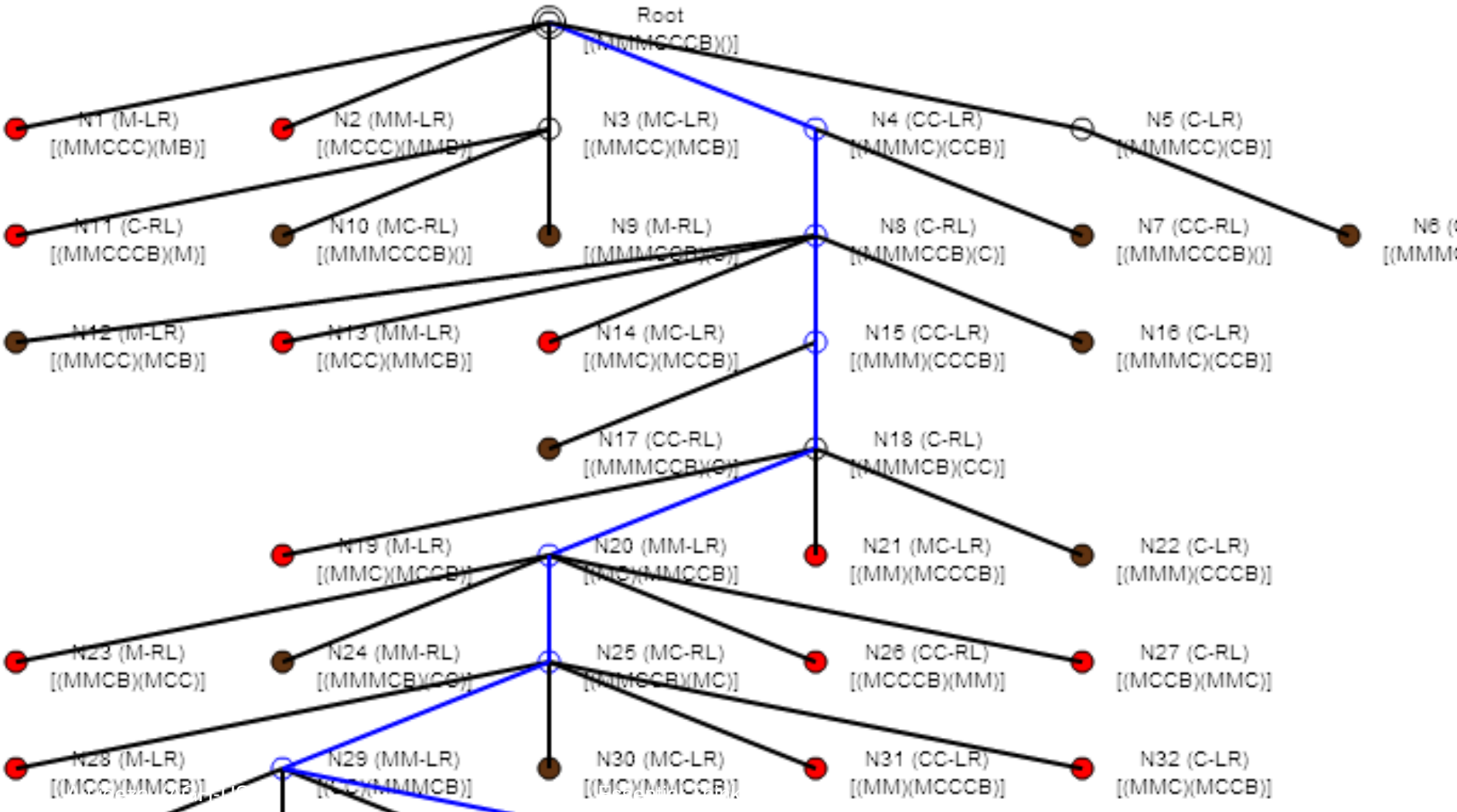
Path Finding



Three generic examples

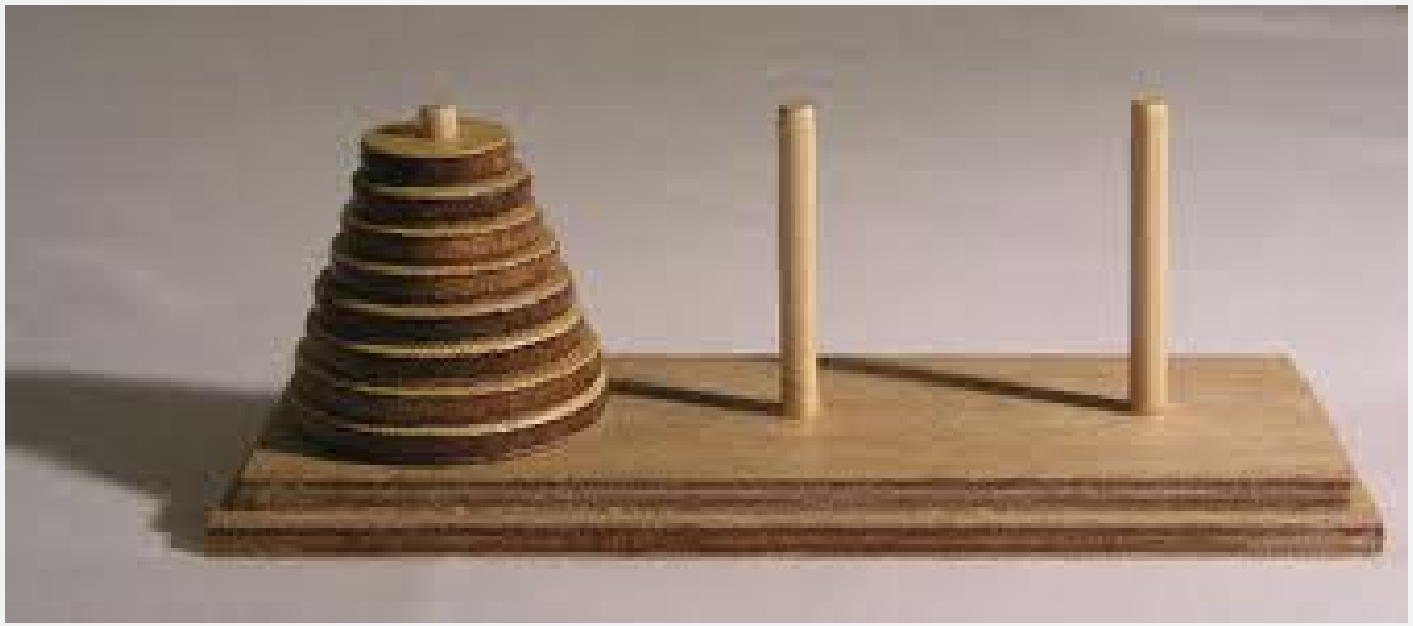
Missionaries and Cannibals

Three generic examples

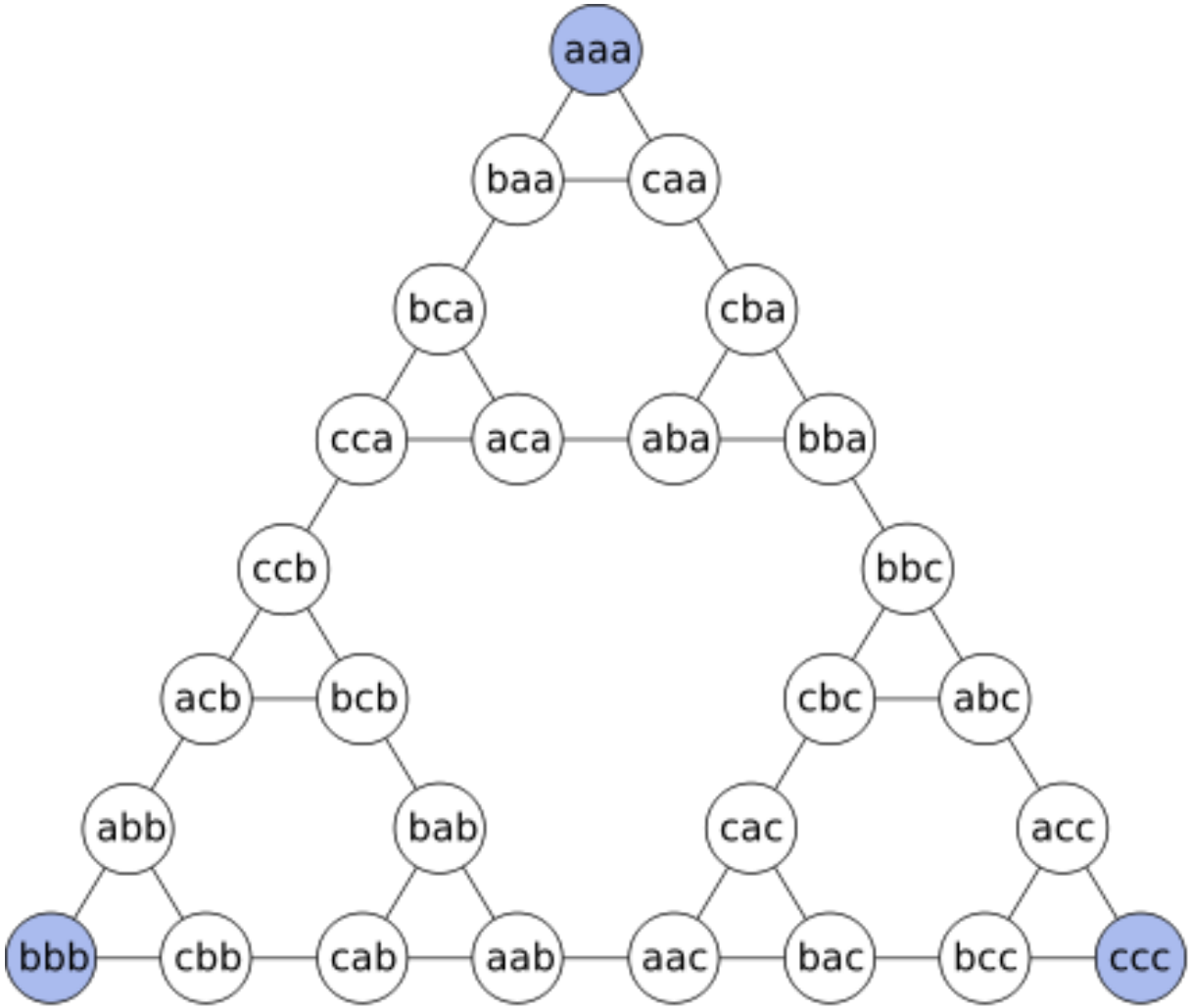


Three generic examples

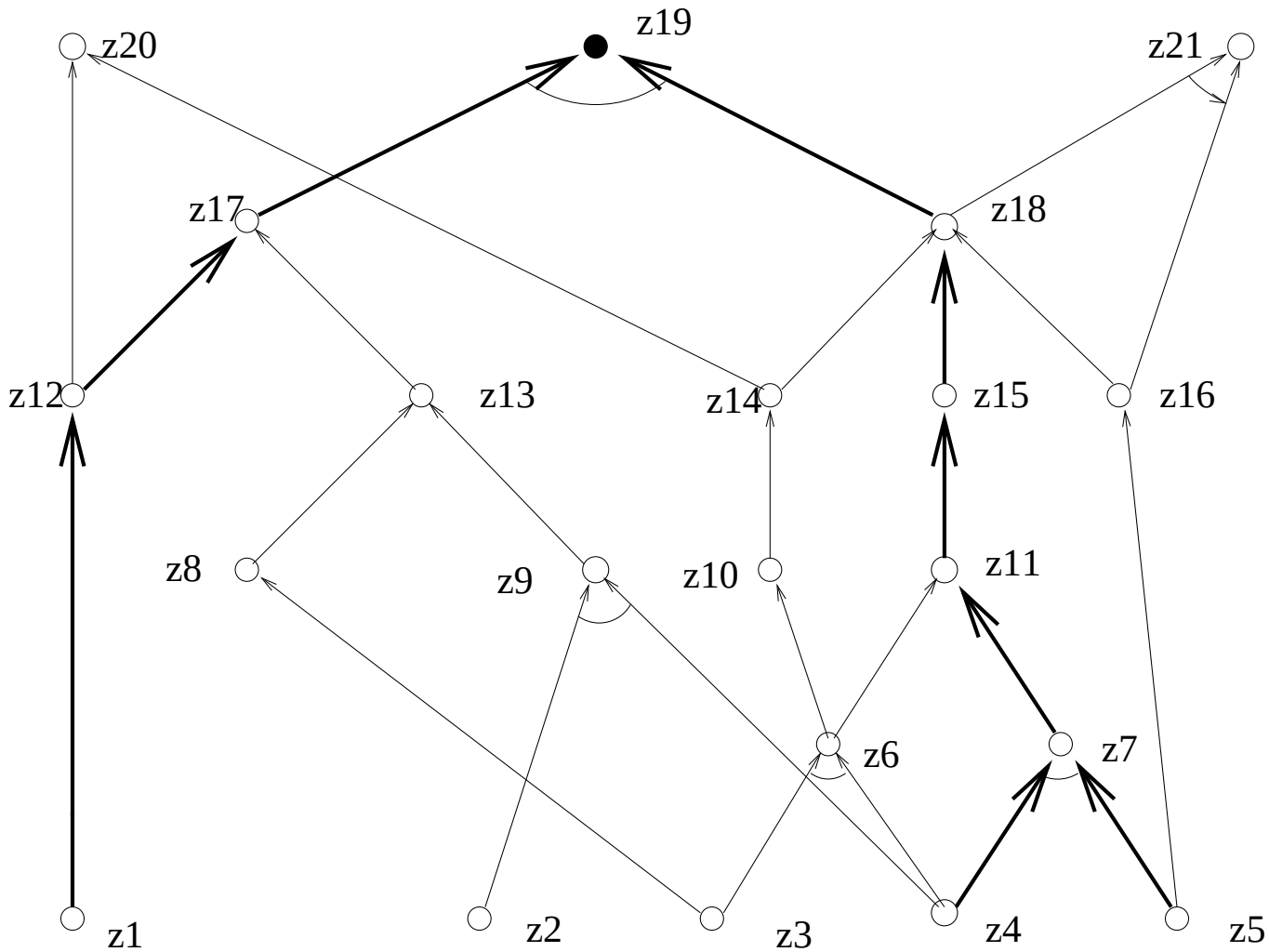
Towers of Hanoi



Three generic examples



Prolog: Generic Tool for Problem Solving



Some example problems

Polynomial –> 0

Does there exists a polynomial function satisfying the following conditions:

- it is always strictly greater than zero,
- for and (small) $\epsilon > 0$, there exists a value of the polinomial less then ϵ
- Proove or
- Disproove.

Two eggs

You are given two eggs. There is a high building of n storeys. An egg dropped from k-th floor brakes; but not for k' < k. Find k in a least number of trials.

Bicycle

In what direction will move a bicycle, when the lower pedal is pulled backwards?

Some example problems

Formula

Find a formla for the sum: $a + aq + aq^2 + aq^3 + ...aq^n$.

Two ships

Two ships move with constant speed. Find the smallest distance.

Counting 1

Write a really fast program for counting 1 in a vector.

Fly problem

Two elephants are approaching each other from opposite direction with constant speed 2 km/h and 3 km/h. Initial distant is 1 km. A fly flies from one to the other of them and again with speed of 20km/h. What distance will it cover until the elephants meet?