PROBLEM SOLVING AND SEARCH IN ARTIFICIAL INTELLIGENCE

Lecture 1

Sarah Gaggl

Dresden, 8th April 2019
What to expect

- The course has 12 lectures, 6 tutorials and a practical part
- Lecture is on Monday in DS 3, 11:10-12:40
- Tutorials are on Monday in DS 2, 9:20-10:50
- Schedule and lecture material will be available at course web-page
  https://iccl.inf.tu-dresden.de/web/Problem_Solving_and_Search_in_Artificial_Intelligence_(SS2019)
- The practical part consists of solving (implementing) a problem and its presentation. Should be performed in groups of two, assignments will be ready at April 15th.
- 3 fixed Dates for practical part (see web-page)
  1) Analysis of the problem, group building
  2) Concept how to solve it
  3) Presentation of solution - last questions
- DEADLINE?
- EXAM?

Literature

- plus additional articles
Agenda

1. Introduction
2. Uninformed Search versus Informed Search (Best First Search, A* Search, Heuristics)
3. Local Search, Stochastic Hill Climbing, Simulated Annealing
4. Tabu Search
5. Answer-set Programming (ASP)
6. Constraint Satisfaction Problems (CSP)
7. Evolutionary Algorithms/ Genetic Algorithms
8. Structural Decomposition Techniques (Tree/Hypertree Decompositions)
What are the Ages of my Three Sons?

Two men meet on the street. One gives the other a puzzle

A: "All three of my sons celebrate their birthday this very day! So, can you tell me how old each of them is?"

B: "Sure, but you’ll have to tell me something about them."

A: "The product of the ages of my sons is 36."

B: "That’s fine but I need more than just this."

A: "The sum of their ages is equal to the number of windows in that building."

B: "Still, I need an additional hint to solve your puzzle."

A: "My oldest son has blue eyes."

B: "Oh, this is sufficient!"
"The product of the ages of my sons is 36."

<table>
<thead>
<tr>
<th>son 1</th>
<th>son 2</th>
<th>son 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
"The sum of their ages is equal to the number of windows in that building."

<table>
<thead>
<tr>
<th>son 1</th>
<th>son 2</th>
<th>son 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
What are the Ages of my Three Sons? ctd.

"The sum of their ages is equal to the number of windows in that building."

\[
\begin{align*}
36 & + 1 & + 1 & = 38 \\
18 & + 2 & + 1 & = 21 \\
12 & + 3 & + 1 & = 16 \\
9 & + 4 & + 1 & = 14 \\
9 & + 2 & + 2 & = 13 \\
6 & + 6 & + 1 & = 13 \\
6 & + 3 & + 2 & = 11 \\
4 & + 3 & + 3 & = 10
\end{align*}
\]
What are the Ages of my Three Sons? ctd.

"The sum of their ages is equal to the number of windows in that building."

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>+</td>
<td>1</td>
<td>+</td>
</tr>
<tr>
<td>18</td>
<td>+</td>
<td>2</td>
<td>+</td>
</tr>
<tr>
<td>12</td>
<td>+</td>
<td>3</td>
<td>+</td>
</tr>
<tr>
<td>9</td>
<td>+</td>
<td>4</td>
<td>+</td>
</tr>
<tr>
<td>9</td>
<td>+</td>
<td>2</td>
<td>+</td>
</tr>
<tr>
<td>6</td>
<td>+</td>
<td>6</td>
<td>+</td>
</tr>
<tr>
<td>6</td>
<td>+</td>
<td>3</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>+</td>
<td>3</td>
<td>+</td>
</tr>
</tbody>
</table>
What are the Ages of my Three Sons? ctd.

"My oldest son has blue eyes."

\[
\begin{array}{ccc}
36 & + & 1 & + & 1 & = & 38 \\
18 & + & 2 & + & 1 & = & 21 \\
12 & + & 3 & + & 1 & = & 16 \\
9 & + & 4 & + & 1 & = & 15 \\
9 & + & 2 & + & 2 & = & 13 \\
6 & + & 6 & + & 1 & = & 13 \\
6 & + & 3 & + & 2 & = & 11 \\
4 & + & 3 & + & 3 & = & 10 \\
\end{array}
\]
What are the Ages of my Three Sons?

Two men meet on the street. One gives the other a puzzle

A: "All three of my sons celebrate their birthday this very day! So, can you tell me how old each of them is?"

B: "Sure, but you’ll have to tell me something about them."

A: "The product of the ages of my sons is 36."

B: "That’s fine but I need more than just this."

A: "The sum of their ages is equal to the number of windows in that building."

B: "Still, I need an additional hint to solve your puzzle."

A: "My oldest son has blue eyes."

B: "Oh, this is sufficient!"

What was difficult on this problem?
Problem Solving

- Where to begin?
- You have to create the plan for generating a solution.
- Always consider all of the available data.
- Can you make connections between the goal and what is given?
Why are Some Problems Difficult to Solve?

• The number of possible solutions in the search space is too large for an exhaustive search.
• The problem is too complicated, and simplified models of the problem are useless.
• The evaluation function of the quality of a solution is noisy or varies with time, which requires an entire series of solutions.
• There are so many constraints that finding even one feasible answer is difficult, let alone searching for an optimal solution.
• The person solving the problem is inadequately prepared.
The Size of the Search Space

Boolean Satisfiability Problem (SAT)

Make a compound statement of Boolean variables evaluate to TRUE.

- For example, consider the following problem of 100 variables given in conjunctive normal form (CNF):

\[ F(x) = (x_{17} \lor \neg x_{37} \lor x_{73}) \land (\neg x_{11} \lor \neg x_{56}) \land \cdots \land (x_{2} \lor x_{43} \lor \neg x_{77} \lor \neg x_{89} \lor \neg x_{97}) . \]

- **Challenge:** find the truth assignment for each variable \( x_i \), for all \( i = 1, \ldots 100 \) s.t. \( F(x) = \text{TRUE} \).
The Size of the Search Space

**Boolean Satisfiability Problem (SAT)**

Make a compound statement of Boolean variables evaluate to **TRUE**.

- For example, consider the following problem of 100 variables given in conjunctive normal form (CNF):

\[ F(x) = (x_{17} \lor \neg x_{37} \lor x_{73}) \land (\neg x_{11} \lor \neg x_{56}) \land \cdots \land (x_{2} \lor x_{43} \lor \neg x_{77} \lor \neg x_{89} \lor \neg x_{97}) \].

- Challenge: find the truth assignment for each variable \( x_i \), for all \( i = 1, \ldots, 100 \) s.t. \( F(x) = \text{TRUE} \).

**Space of possible solutions.**

- Any binary string of length 100 is a possible solution.
- Two choices for each variable, and taken over 100 variables, generates \( 2^{100} \) possibilities.
The Size of the Search Space ctd.

- Size of the search space $S$ is $|S| = 2^{100} \approx 10^{30} = 1\,000\,000\,000\,000\,000\,000\,000\,000\,000\,000\,000$.
- The number of bacterial cells on Earth is estimated at around $5 \times 10^{30}$.
- If we had a computer that could test 1000 strings per second and could have started at the beginning of time itself, 15 billion years ago (Big Bang!) we would have examined fewer than 1% of all the possibilities by now!
- Trying out all alternatives is out of the question.
- Choice of which evaluation function to use.
- Solutions closer to the right answer should yield better evaluations than those who are far away.
- If we try a string $x$ and $F(x)$ returns TRUE, we are done. But what if $F(x)$ returns FALSE?
- How to find a function which gives more than just "right" or "wrong"?
Traveling Salesperson Problem (TSP)

- Given $n$ cities and the distances between each pair of cities;
- Traveling salesperson must visit every city exactly once and return home covering the shortest distance.
The Size of the Search Space ctd.

**Traveling Salesperson Problem (TSP)**

- Given \( n \) cities and the distances between each pair of cities;
- Traveling salesperson must visit every city exactly once and return home covering the shortest distance.

**Search Space**

- Set of permutations of \( n \) cities.
- \( 2n \) different ways (for symmetrical TSP) to represent one tour.
- There are \( n! \) ways to permute \( n \) numbers.
- \(|S| = n!/(2n) = (n - 1)!/2\)
The Size of the Search Space ctd.

- $|\mathcal{S}| = n!/(2n) = (n - 1)!/2$
- For any $n > 6$, number of possible solutions to the TSP with $n$ cities is larger than the number of possible solutions to the SAT problem with $n$ variables.
- For $n = 6$: $5!/2 = 60$ solutions to the TSP and $2^6 = 64$ solutions to a SAT.
- For $n = 7$: 360 solutions to the TSP and 128 to the SAT.
- Search space increases very quickly with increasing $n$.
- A 50-city TSP has more solutions than existing liters of water on the planet.
- However, the evaluation function for the TSP is more straightforward than for SAT.
- Table with distances between each pair of cities.
- After $n$ addition operations we could calculate the distance of any candidate tour and use this to evaluate its merit.
- $cost = dist(15, 3) + dist(3, 11) + \cdots + dist(6, 15)$
Modeling the problem

- We only find the solution to a model of the problem.
- All models are simplifications of the real world.
- Problem $\rightarrow$ Model $\rightarrow$ Solution
  1. Use an approximate model of a problem and find the precise solution: Problem $\rightarrow$ Model$_a$ $\rightarrow$ Solution$_p$(Model$_a$)
  2. Use a precise model of the problem and find an approximate solution: Problem $\rightarrow$ Model$_p$ $\rightarrow$ Solution$_a$(Model$_p$)
- Which one is better?
Modeling the problem

- We only find the solution to a model of the problem.
- All models are simplifications of the real world.
- Problem $\rightarrow$ Model $\rightarrow$ Solution
  1. Use an approximate model of a problem and find the precise solution: Problem $\rightarrow$ Model$_a$ $\rightarrow$ Solution$_p$(Model$_a$)
  2. Use a precise model of the problem and find an approximate solution: Problem $\rightarrow$ Model$_p$ $\rightarrow$ Solution$_a$(Model$_p$)
- Which one is better?
- Solution$_a$(Model$_p$) is better than Solution$_p$(Model$_a$).
Change over time

Problems my change

- before you model them,
- while you derive a solution, and
- after you execute the solution.

TSP - Travel time between two cities depends on many factors:

- traffic lights
- slow-moving trucks
- flat tire
- weather
- many more...

TU Dresden, 8th April 2019

PSSAI slide 21 of 29
Constraints

- Almost all practical problems pose constraints
- Two types of constraints:
  - Hard constraints, and
  - Soft constraints.
- Constraints make the search space smaller, but
  - It is hard to create operators that will act on feasible solution and generate in turn new feasible solutions that are an improvement of previous solution.
  - The geometry of search space gets tricky.
Constraints ctd.

Timetable of the classes at a college in one semester

We are given

- list of **courses** that are offered;
- list of **students** assigned to each class;
- **professors** assigned to each class;
- list of available **classrooms**, and information for size and other facilities that each offer.
Timetable of the classes at a college in one semester

We are given

- list of courses that are offered;
- list of students assigned to each class;
- professors assigned to each class;
- list of available classrooms, and information for size and other facilities that each offer.

Construct timetables that fulfill hard constraints:

- Each class must be assigned to an available room that has enough seats and requisite facilities.
- Students who are enrolled in more than one class cannot have their classes held at the same time on the same day.
- Professors cannot be assigned to teach courses that overlap in time.
Timetable - Soft Constraints:

- Courses that meet **twice a week** should preferably be assigned to Mondays and Wednesdays or Tuesdays and Thursdays.
- Courses that meet **three times per week** should preferably be assigned to Mondays, Wednesdays, and Fridays.
- Course time should be assigned so that students do **not** have to take final exams for multiple courses without any break in between.
- If more than one room satisfies the requirements for a course and is available at the designated time, the course should be assigned to the room with the **capacity that is closest to the class size**.
Timetable - Soft Constraints:

- Courses that meets twice a week should preferably be assigned to Mondays and Wednesdays or Tuesdays and Thursdays.
- Courses that meets three times per week should preferably be assigned to Mondays, Wednesdays, and Fridays.
- Course time should be assigned so that students do not have to take final exams for multiple courses without any break in between.
- If more than one room satisfies the requirements for a course and is available at the designated time, the course should be assigned to the room with the capacity that is closest to the class size.

- Any timetable that meets the hard constraints is feasible.
- The timetable has to be optimized in the light of soft constraints.
- Each soft constraint has to be quantified.
- We can evaluate two candidate assignments and decide that one is better than other.
Solve the Problem!

- Mr. Smith and his wife invited four other couples for a party.
- When everyone arrived, some of the people in the room shook hands with some of the others.
- Nobody shook hands with their spouse and nobody shook hands with the same person twice.
- After that, Mr. Smith asked everyone how many times they shook someone’s hand.
- He received different answers from everybody.
- How many times did Mrs. Smith shake someone’s hand?
Summary

Problem solving is difficult for several reasons:

- Complex problems often pose an enormous number of possible solutions.
- To get any sort of solution at all, we often have to introduce simplifications that make the problem tractable. As a result, the solutions that we generate may not be very valuable.
- The conditions of the problem change over time and might even involve other people who want to fail you.
- Real-world problems often have constraints that require special operations to generate feasible solutions.
References