Artificial Intelligence

MA course 2022/2023

Università della Svizzera italiana

Scuola universitaria professionale della Svizzera italiana

SUPSI

Istituto Dalle Molle di studi sull'intelligenza artificiale

USI/SUPSI IDSIA

Tutorial 6

Midterm preparation

Giorgia Adorni, Umberto Junior Mele

Dalle Molle Institute for Artificial Intelligence, Switzerland

giorgia.adorni@usi.ch
umberto.junior.mele@usi.ch

Topics

- **Metaheuristics**
- **Simulated Annealing**
- **Iterated Local Search**
- **Genetic Algorithms**
- **Combinatorial Optimisation Problems**

Metaheuristics

1. Explain what metaheuristics are and how metaheuristics search works.

Metaheuristics

1. Explain what metaheuristics are and how metaheuristics search works.

Make a brief introduction to *combinatorial optimisation problems*.
Mention that to solve these complex problems (NP-hard) are used practical solutions able to solve them in *polynomial time*.

There are two main approaches:

1) **Exact methods** (search algorithms usually based on linear integer programming formulation and branch&bound) that guarantee to find an *optimal solution*, but they only apply to the problem of small size or require a *long computational time*.

2) **Approximation algorithms** try to compute in a *short time* a *solution* that is as *close* as possible *to the optimal one*. There is no guarantee about the solution quality. *Metaheuristics* belong to this set of methods.

Metaheuristics

1. **Explain what metaheuristics are and how metaheuristics search works.**

In **metaheuristic search**, we start from a *solution* (or a set of solutions) that is (are) *iteratively modified* using *stochastic processes* (usually non-deterministic).
Previous results are used to update the search and to generate new, better solutions.

This is an optimisation procedure.
The search space (neighbourhood) is efficiently explored to find good (near-optimal) feasible solutions.
They provide no guarantee of global or local optimality.
They can accept worse solutions during the search.
They can be used in conjunction with exact methods.
Usually, incorporate mechanisms to avoid getting trapped in confined areas of the search space.

Metaheuristics

1. **Explain what metaheuristics are and how metaheuristics search works.**

```
Meta Heuristic search
    input
    initial solution s_start (or an initial set of solutions)
    objective function f
    neighborhood function N

    current ← s_start (current should also be a set of solutions)
    while terminal condition is met
        stochastically compute a solution next ∈ N(current)
        Following a criterium decide whether or not
            to continue the search from next
            by setting current ← next
    end while
    output current
```

Topics

- Metaheuristics
- **Simulated Annealing**
- Iterated Local Search
- Genetic Algorithms
- Combinatorial Optimisation Problems

Simulated Annealing

1. **Show the Simulated Annealing algorithm representation and explain it.**
2. **States which are the advantages and disadvantages of this approach.**

Simulated Annealing

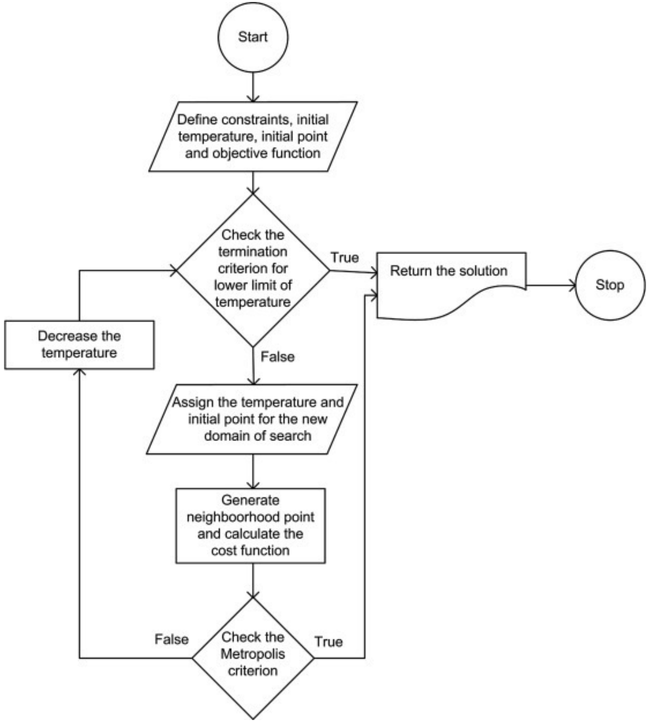
1. **Show the Simulated Annealing algorithm representation and explain it.**

```
Simulated Annealing(problem) return a solution
    T ← determine a starting temperature
    current ← generate an initial solution
    best ← current
    While not yet frozen do
       While not yet at equilibrium for this temperature do
            next ← a random solution selected from Neigh(current)
            ΔE ← f(next) - f(current)
            if ΔE<0 then current ← next
                          if f(next) < f(best) then best ← next
                   else
                   choose a random number r uniformly from [0.1]
                   if r< e^(-ΔE/T) then current ← next

        end while
           lower the temperature T
    end while
    Return best
```

Simulated Annealing

1. **Show the Simulated Annealing algorithm representation and explain it.**



Simulated Annealing

1. **Show the Simulated Annealing algorithm representation and explain it.**

Generate an initial solution (feasible or not)
Set the initial temperature and the decrease factor
Store the best solution so far

Iterate:
Generate a new solution from the neighbourhood
The current solution is always replaced by a new one if this modification reduces the objective value, while a modification increasing the objective value Δ is only accepted with a probability $e^{-\frac{\Delta}{T}}$ (at a high temperature, the probability of accepting an increase to the objective value is high).
Lower the remperature

Simulated Annealing

1. Show the Simulated Annealing algorithm representation and explain it.
2. States which are the advantages and disadvantages of this approach.

Simulated Annealing

2. States which are the advantages and disadvantages of this approach.

Advantages:
Easy implementation
Possibility of finding a global optimal even after finding a local minimum
It can deal with noisy data and highly non-linear models
Statistically guarantees to find an optimal solution

Disadvantages:
A lot of parameters have to be tuned
The precision of the numbers used in its implementation has a significant effect on results' quality
There is a tradeoff between the quality of the result and the time taken for the algorithm to run
Repeatedly annealing with a schedule is very slow, especially if the cost function is expensive to compute

Topics

- Metaheuristics
- Simulated Annealing
- **Iterated Local Search**
- Genetic Algorithms
- Combinatorial Optimisation Problems

Iterated Local Search

1. **Explain how Iterated Local Search works**

Iterated Local Search

1. **Explain how Iterated Local Search works.**

Algorithm 1 Iterated Local Search

$s_0 = \mathsf{GenerateInitialSolution}$
$s^* = \mathsf{LocalSearch}(s_0)$
repeat
  $s' = \mathsf{Perturbation}(s^*, \mathit{history})$
  $s^{*\prime} = \mathsf{LocalSearch}(s')$
  $s^* = \mathsf{AcceptanceCriterion}(s^*, s^{*\prime}, \mathit{history})$
until termination condition met

Iterated Local Search

1. Explain how Iterated Local Search works.

Iteratively build a sequence of solutions generated by an embedded heuristic, leading to far better solutions than if one were to use repeated random trials of that heuristic.

- Start with a random solution or one returned by some greedy construction heuristic
- Use a local search algorithm to obtain a local optimum
- Perturb the solution (with a certain strength) and reapply local search
- Accept the new solution according to some criterion (intensification & diversification trade-off)

The potential power of iterated local search lies in its biased sampling of the set of local optima.

The efficiency of this sampling depends both on the kinds of perturbations and on the acceptance criteria.

Topics

- Metaheuristics
- Simulated Annealing
- Iterated Local Search
- **Genetic Algorithms**
- Combinatorial Optimisation Problems

Genetic Algorithms

1. **Describe what Genetic Algorithms are.**
2. **Describe the Roulette Wheel parent selection technique.**
3. **What are some disadvantages of Genetic Algorithms?**
4. **What is the difference between Mutation and Crossover?**

Genetic Algorithms

1. **Describe what Genetic Algorithms are.**

GAs are Adaptive metaheuristics methods used to solve search and optimisation problems.

They are based on the genetic processes of biological organisms: over many generations, natural populations evolve according to the principles of natural selection and survival of the fittest.

Given a population of **individuals**, each representing a possible **solution** to a given problem, each individual is assigned a **fitness score** according to how good a solution to the problem it is.
The highly fit individuals can **reproduce** and **cross breeding** with other individuals. This produces new individuals as **offspring**, which share features taken from each parent. The least fit members of the population are less likely to be selected for reproduction and **die out**.

Genetic Algorithms

1. **Describe what Genetic Algorithms are.**

```
Procedure GA
begin
      t ← 0
      initialize population P(t) with m individuals
      evaluate population P(t)
      while termination condition is not met
         begin
                  select parents from P(t)
                  generates new individuals using reproduction rules
                  some individuals die in P(t)
                  form a new population P(t+1)
                  evaluate population P(t+1)
                  t ←  t + 1
         end
return the best individual
end
```

Genetic Algorithms

1. Describe what Genetic Algorithms are.
2. **Describe the Roulette Wheel parent selection technique.**
3. What are some disadvantages of Genetic Algorithms?
4. What is the difference between Mutation and Crossover?

Genetic Algorithms

2. Describe the Roulette Wheel parent selection technique.

Selection process to pick the individuals from the current population that will be used as parents for the next generation.
Each individual is given a chance to become a parent in proportion to their fitness. It is called roulette wheel selection as the chances of selecting a parent can be seen as spinning a roulette wheel with the size of the slot for each parent being proportional to its fitness. Obviously, those with the largest fitness (slot sizes) have more chance of being chosen.

Roulette wheel selection can be implemented as follows:

- Sum the fitnesses of all the population members. Call this TF (total fitness).
- Generate a random number n, between 0 and TF.
- Return the first population member whose fitness added to the preceding population members is greater than or equal to n.

Genetic Algorithms

1. Describe what Genetic Algorithms are.
2. Describe the Roulette Wheel parent selection technique.
3. **What are some disadvantages of Genetic Algorithms?**
4. What is the difference between Mutation and Crossover?

Genetic Algorithms

3. What are some disadvantages of Genetic Algorithms?

No guarantee of finding global maxima.
The likelihood of getting stuck in a local maximum early on is something you might have to deal with, for example, with some kind of simulated annealing mutation rate decay.

Time is taken for convergence. You usually need a decent-sized population and a lot of generations before you see good results.

Binary coding of genetics can be problematic. The standard algorithm codes the genes as binary strings similar to how DNA codes information in living organisms. This is fine if the parameters really are binary. However, if the parameters are real numbers, then the natural distances between points are destroyed in the transformation.

Genetic Algorithms

3. What are some disadvantages of Genetic Algorithms?

Hyperparameter tuning. Fine-tuning all the parameters for the GA, like mutation rate, elitism percentage, crossover parameters, fitness normalization/selection parameters, etc, is often just trial and error.

There is a risk of Premature Convergence. This means that the parents are not able to generate offspring that are superior to their parents.

Over-fitting. Since you cannot generate all the possible permutations of holes and available blocks, you are bound to have over-fitting with your sample data using unguided mutation. If you do not train the algorithm long enough (generations) then the usability would be very low.

Genetic Algorithms

1. Describe what Genetic Algorithms are.
2. Describe the Roulette Wheel parent selection technique.
3. What are some disadvantages of Genetic Algorithms?
4. **What is the difference between Mutation and Crossover?**

Genetic Algorithms

4. **What is the difference between Mutation and Crossover?**

Mutation is an operation that is applied to a single individual in the population without making use of gene values in any other individuals. It can e.g. introduce some noise in the chromosome. For example, if the chromosomes are binary, a mutation may simply be the flip of a bit (or gene).

Crossover is an operation which takes as input two individuals (often called the "parents") and somehow combines their chromosomes, so as to produce usually two other chromosomes (the "children"), which inherit, in a certain way, the genes of both parents.

Topics

- **Metaheuristics**
- **Simulated Annealing**
- **Iterated Local Search**
- **Genetic Algorithms**
- **Combinatorial Optimisation Problems**

Combinatorial Optimisation Problems

1. **Give the general definition of the Knapsack problem.**
2. **Compare the Dynamic Programming approach with Greedy Search Algorithm to solve the Knapsack problem.**

Combinatorial Optimisation Problems

1. **Give the general definition of the Knapsack problem.**

We are given $n > 0$ distinct items and a knapsack of capacity $\mathrm{C} > 0$.

For each item i, let $w_i > 0$ and $p_i > 0$ denote its weight and profit.

The goal is to select a subset of items such that:

- the **profit** is maximised $\sum_{i=1}^{n} p_i x_i$
- the **capacity** of the knapsack is not exceeded $\sum_{i=1}^{n} w_i x_i \leq C$

where $x_i \in \{0, 1\}$ for $i \in \{1, \dots, n\}$

Each item cannot be put into the knapsack more than once or be partially included in the knapsack.

Combinatorial Optimisation Problems

1. Give the general definition of the Knapsack problem.
2. Compare the Dynamic Programming approach with Greedy Search Algorithm to solve the Knapsack problem.

Combinatorial Optimisation Problems

2. Compare the Dynamic Programming approach with Greedy Search Algorithm to solve the Knapsack problem.

Greedy Search

Approximation method, Time complexity: $O(n \, log \, n)$

Procedure:

1. Calculate profit/weight;
2. Sort the items, in descending order, according to profit/volume;
3. Insert one item after the other, and discard it if it overfills the capacity.

Combinatorial Optimisation Problems

2. Compare the Dynamic Programming approach with Greedy Search Algorithm to solve the Knapsack problem.

<u>Dynamic programming</u>

Exact method, Time complexity: $O(n * C)$
Procedure:

1. Create a 2D array (Table) of n+1 rows (items) and C+1 columns (weights);
2. The profit of item *i* and weight *j* depend if it is included or not in the knapsack:
   i. The maximum profit without item *i* is at row *i-1*, column *j*.
   ii. The maximum profit with item *i*, is the profit of item *i* itself + the maximum profit that can be obtained with the remaining capacity of the knapsack.

Combinatorial Optimisation Problems

2. **Compare the Dynamic Programming approach with Greedy Search Algorithm to solve the Knapsack problem.**

<u>Dynamic programming</u>

…

3. The total profit is in the last row and column of the table: $Table[n, C]$;
4. Use backtracking from cell $Table[n, C]$ to see which elements are included in the table:
    a. go one level up and check whether the value in the upper level is smaller or not
        i. If YES, then it means that the difference is caused because of including the last item. Now subtract the weight of the last (selected) item from *j* and repeat the same process for the resultant value of the *j-th* column.
        ii. If NO, then go one level up more and check for the difference in the value. Keep going up until you see the difference in the value.