EUNICE EUROPEAN UNIVERSITY

EUNICE RESEARCH SUMMER SCHOOL 2025

APPLICATIONS OF AI

Introduction to Evolutionary Computation

Maciej Komosinski Konrad Miazga

Institute of Computing Science
Poznan University of Technology
maciej.komosinski@cs.put.poznan.pl

Co-funded by the European Union

Verifying questions – do you know...

- Can optimization be used for **maximization**, or only for minimization?

Verifying questions – do you know. . .

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?

Verifying questions – do you know...

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?

Verifying questions – do you know. . .

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?

Verifying questions – do you know...

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?
- What is **NP-hard**?

Verifying questions – do you know...

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?
- What is **NP-hard**?
- What is **local search**?

Verifying questions – do you know. . .

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?
- What is **NP-hard**?
- What is **local search**?
- What are the main steps in an **evolutionary algorithm**?

Verifying questions – do you know. . .

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?
- What is **NP-hard**?
- What is **local search**?
- What are the main steps in an **evolutionary algorithm**?
- What is the role of **mutation** and **crossover** in evolutionary algorithms?

Optimization?

- How to find good (best) solutions (to improve quality, time, cost...)
- In hard problems

Optimization?

- How to find good (best) solutions (to improve quality, time, cost. . .)
- In hard problems

- without the need to be an expert in solving each particular problem and in its domain

Components of an optimization problem

- Problem

Components of an optimization problem

- Problem
- An instance of a problem

Components of an optimization problem

- Problem
- An instance of a problem
- Solution space

Components of an optimization problem

- Problem
- An instance of a problem
- Solution space
- Constraints

Components of an optimization problem

- Problem
- An instance of a problem
- Solution space
- Constraints
- Evaluation criteria

Components of an optimization problem

- Problem
- An instance of a problem
- Solution space
- Constraints
- Evaluation criteria
- Goal function

Components of an optimization problem

- Problem
- An instance of a problem
- Solution space
- Constraints
- Evaluation criteria
- Goal function
- Algorithm

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)
- Solution space: All combinations of amounts invested, x_1, x_2, x_3, x_4

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)
- Solution space: All combinations of amounts invested, x_1, x_2, x_3, x_4
- Constraints: $x_1 + x_2 + x_3 + x_4 \leq$ €100k and each $x_i \geq 0$

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)
- Solution space: All combinations of amounts invested, x_1, x_2, x_3, x_4
- Constraints: $x_1 + x_2 + x_3 + x_4 \leq$ €100k and each $x_i \geq 0$
- Evaluation criteria:
  Expected gain (maximize), variance of the portfolio return (minimize)

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)
- Solution space: All combinations of amounts invested, x_1, x_2, x_3, x_4
- Constraints: $x_1 + x_2 + x_3 + x_4 \leq$ €100k and each $x_i \geq 0$
- Evaluation criteria:
  Expected gain (maximize), variance of the portfolio return (minimize)
- Goal function: Maximize (gain − risk)

Components of an optimization problem: example

- Problem:
  Invest M money in K assets, expected gain of each is G_k, risk of each is R_k
- An instance of a problem:
  Invest €100k in 4 stocks, gains (1.01, 1.03, 1.02, 1.11), risks (0.7, 0.54, 0.6, 0.95)
- Solution space: All combinations of amounts invested, x_1, x_2, x_3, x_4
- Constraints: $x_1 + x_2 + x_3 + x_4 \leq$ €100k and each $x_i \geq 0$
- Evaluation criteria:
  Expected gain (maximize), variance of the portfolio return (minimize)
- Goal function: Maximize (gain $-$ risk)
- Algorithm: ...

What are *combinatorial* optimization problems?

(as opposed to *numerical* optimization problems)

- Many real world problems can be expressed as a combination of elements
  - e.g., Traveling Salesman Problem is a permutation of integers

What are *combinatorial* optimization problems?

(as opposed to *numerical* optimization problems)

- Many real world problems can be expressed as a combination of elements
  - e.g., Traveling Salesman Problem is a permutation of integers
- More formally:
  - minimizing (or maximizing)
  - a function $f(x)$
  - subject to constraints, e.g., $g(x) > 0$
  - x is a solution, often encoded with vectors of symbols

What are *combinatorial* optimization problems?

(as opposed to *numerical* optimization problems)

- Many real world problems can be expressed as a combination of elements
  - e.g., Traveling Salesman Problem is a permutation of integers
- More formally:
  - minimizing (or maximizing)
  - a function $f(x)$
  - subject to constraints, e.g., $g(x) > 0$
  - x is a solution, often encoded with vectors of symbols
- NP-hard!

Computational complexity

- Time and memory complexity – examples

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | ...
  - selection of a subset of features:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path:

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path: A B C D E | B A C D E | D E B A C | . . .

Computational complexity

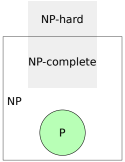
- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path: A B C D E | B A C D E | D E B A C | . . .
- Plot: number_of_solutions(n)

Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path: A B C D E | B A C D E | D E B A C | . . .
- Plot: number_of_solutions(n)
- Plot: time_to_find_optimum(n)

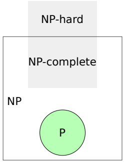
Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | ...
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | ...
  - shortest path: A B C D E | B A C D E | D E B A C | ...
- Plot: number_of_solutions(n)
- Plot: time_to_find_optimum(n)
- Complexity classes:
  P (easy to find the optimum, easy to verify),
  NP (easy to verify),
  NP-C,
  strongly NP-C,
  ...



Computational complexity

- Time and memory complexity – examples
  - pair comparison/confrontation: ○ ○ ○ ○ ● ● ○ | ○ ○ ○ ● ○ ● ○ | ○ ● ○ ○ ○ ○ ● | . . .
  - selection of a subset of features: ○ ● ○ ○ ○ ● ● | ● ○ ● ● ○ ● ○ | ● ● ○ ● ● ○ ● | . . .
  - shortest path: A B C D E | B A C D E | D E B A C | . . .
- Plot: number_of_solutions(n)
- Plot: time_to_find_optimum(n)
- Complexity classes:
  P (easy to find the optimum, easy to verify),
  NP (easy to verify),
  NP-C,
  strongly NP-C,
  . . .



- The simplest algorithms: a heuristic, random, exhaustive

A large number of possible solutions

For example, the Boolean satisfiability problem (SAT):

$$F(x) = (x_{13} \vee \overline{x}_{23} \vee x_{34}) \wedge (\overline{x}_{13} \vee x_{23} \vee \overline{x}_{34}) \wedge \ldots = TRUE$$

For 100 variables (two possible values per variable)

$|S| =$

A large number of possible solutions

For example, the Boolean satisfiability problem (SAT):

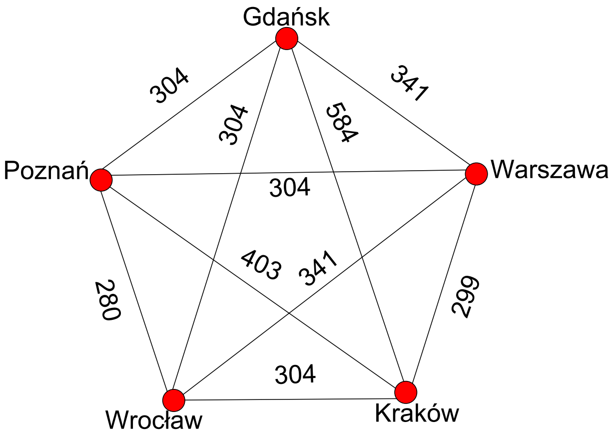
$$F(x) = (x_{13} \vee \overline{x}_{23} \vee x_{34}) \wedge (\overline{x}_{13} \vee x_{23} \vee \overline{x}_{34}) \wedge \ldots = TRUE$$

For 100 variables (two possible values per variable)

$|S| = 2^{100} \approx 10^{30}$

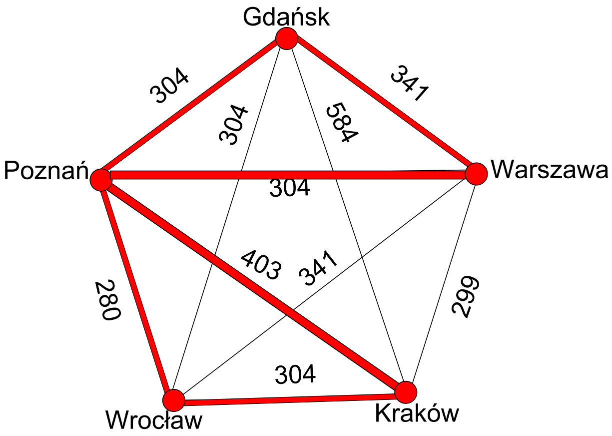
Traveling salesman problem (TSP)

http://www.alife.pl/files/opt/d/tsp-js/tsp.html?lang=en



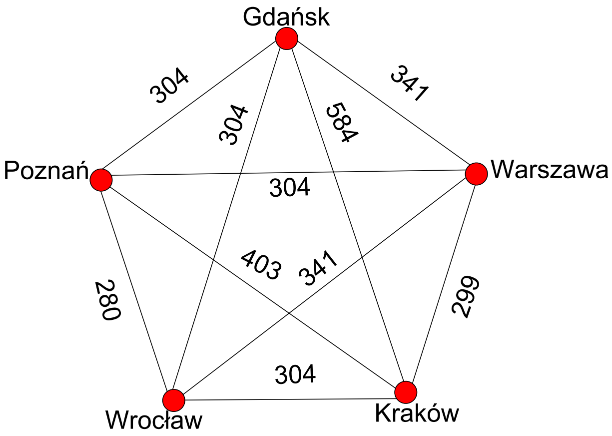
Traveling salesman problem (TSP)

http://www.alife.pl/files/opt/d/tsp-js/tsp.html?lang=en



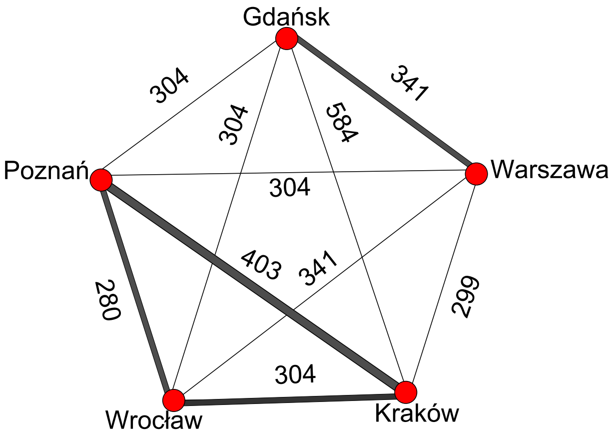
Traveling salesman problem (TSP)

http://www.alife.pl/files/opt/d/tsp-js/tsp.html?lang=en



Traveling salesman problem (TSP)

http://www.alife.pl/files/opt/d/tsp-js/tsp.html?lang=en



Traveling salesman problem (TSP) and NP-hardness

$$|S| = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

Each route is expressed in $2n$ different ways,
$n!$ ways of permutation of n items

10 cities	181 440
20 cities	60 822 550 204 416 000
50 cities	304 251 932 017 133 780 436 126 081 660 647 688 443 776 415 689 605 120 000 000 000

$$|S| = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

Each route is expressed in $2n$ different ways,
$n!$ ways of permutation of n items

10 cities	181 440
20 cities	60 822 550 204 416 000
50 cities	304 251 932 017 133 780 436 126 081 660 647 688 443 776 415 689 605 120 000 000 000

For comparison: selection of a subset

n=10	1 024
n=20	1 048 576
n=50	1 125 899 906 842 624

Traveling salesman problem (TSP) and NP-hardness

$$|S| = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

Each route is expressed in $2n$ different ways,
$n!$ ways of permutation of n items

10 cities	181 440
20 cities	60 822 550 204 416 000
50 cities	304 251 932 017 133 780 436 126 081 660 647 688 443 776 415 689 605 120 000 000 000

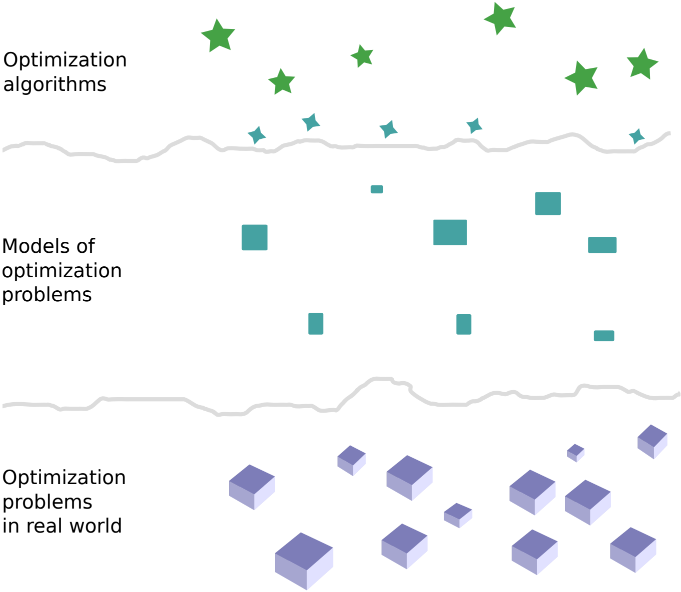
For comparison: selection of a subset

n=10	1 024
n=20	1 048 576
n=50	1 125 899 906 842 624

For comparison: evaluation of pairs

n=10	45
n=20	190
n=50	1 225

Problems, models, instances, algorithms

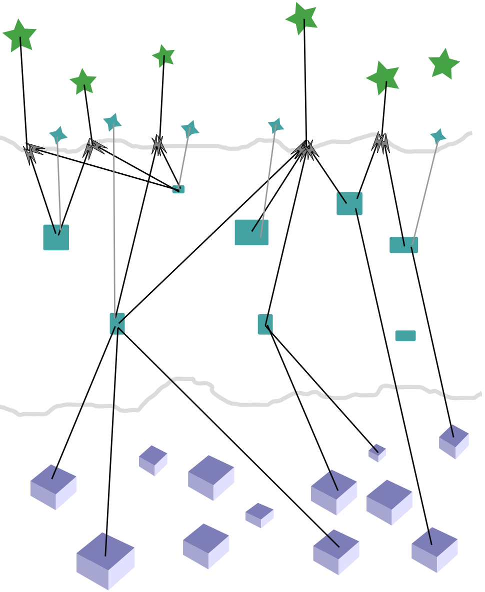


Problems, models, instances, algorithms – relationships

Optimization algorithms

Models of optimization problems

Optimization problems in real world



Motivations for developing optimization methods

- Economy
- Machine vision
- Molecular biology
- Optimal power flow
- Structural optimization
- Robotics
- Database systems
- Computer graphics

- Medicine
- Telecommunications
- Artificial intelligence
- Integrated circuit design automation
- Computer architecture design
- Computer networks
- Image processing
- Security

Motivations for developing optimization methods

- Economy
- Machine vision
- Molecular biology
- Optimal power flow
- Structural optimization
- Robotics
- Database systems
- Computer graphics
- Medicine
- Telecommunications
- Artificial intelligence
- Integrated circuit design automation
- Computer architecture design
- Computer networks
- Image processing
- Security

NP-hard.

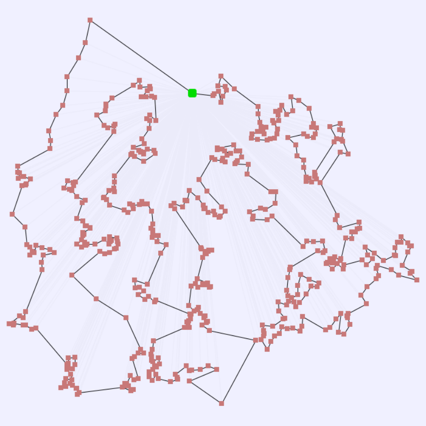
Sample applications of optimization methods

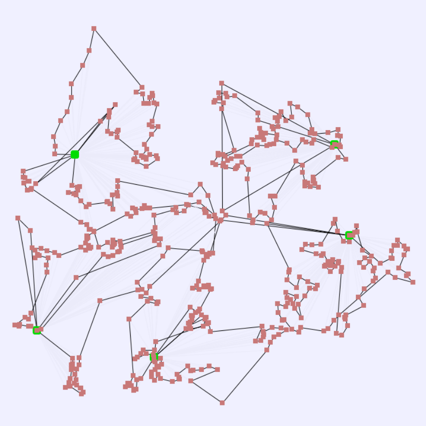
- Design of electronic circuits (VLSI)
- Telecommunication network design
- Knowledge discovery / Machine Learning
- Neural network training & design
- Automatic control
- Business scheduling and planning
- Games
- Self-adapting computer programs
- Test-data generation
- Medical image analysis
- DNA Sequencing

Examples of NP-hard problems from my experience (1/6)

OptiFacility, http://optifacility.mooncoder.com/

Logistics, transport, suppliers, routes...

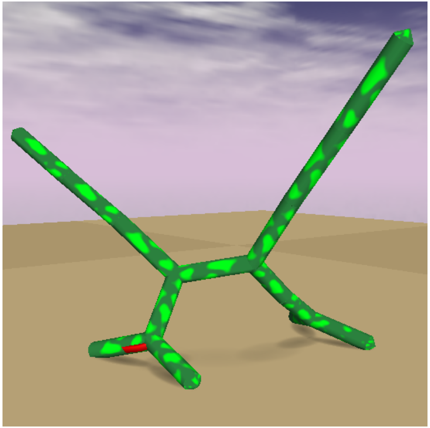


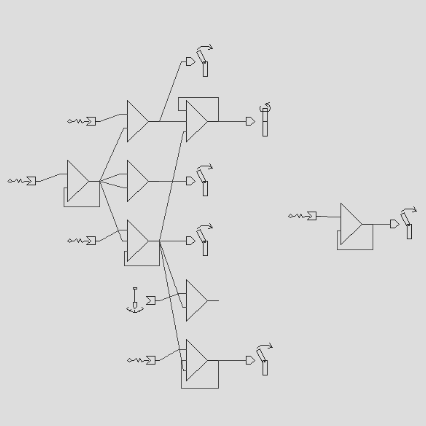


Examples of NP-hard problems from my experience (2/6)

Framsticks, http://www.framsticks.com/

Designs, control systems, neural networks...

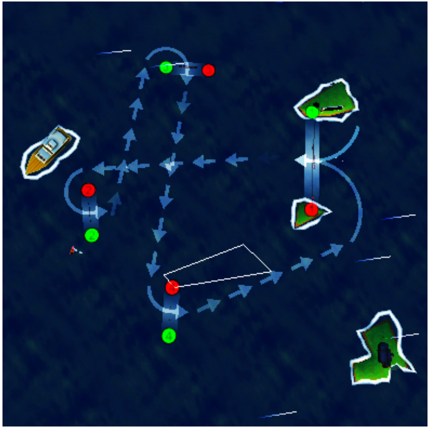


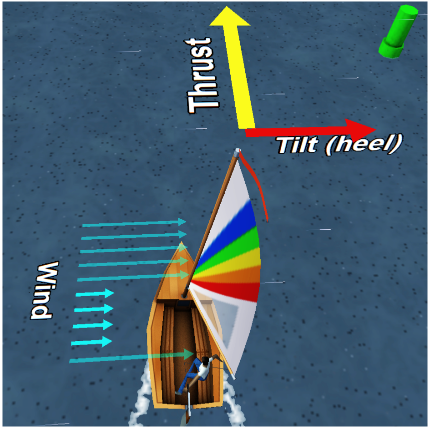


Examples of NP-hard problems from my experience (3/6)

Top Sailor, http://sailor.mooncoder.com/

Route planning, control, strategy...





Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text
- Select some of the office staff to transfer to another location

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text
- Select some of the office staff to transfer to another location
- Choose the best route for the metro line and its stops

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text
- Select some of the office staff to transfer to another location
- Choose the best route for the metro line and its stops
- Suggest the best order in which cars are assembled on the assembly line

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text
- Select some of the office staff to transfer to another location
- Choose the best route for the metro line and its stops
- Suggest the best order in which cars are assembled on the assembly line
- Provide a schedule for shutting down and maintenance of the power plant

Practical examples of NP-hard problems (4/6)

- Create groups of people according to their competencies
- Assign offices to locations in a building
- Determine the route for the garbage truck
- Draw the most readable diagram/graph based on the specification
- Decide where to place pictures in the text
- Select some of the office staff to transfer to another location
- Choose the best route for the metro line and its stops
- Suggest the best order in which cars are assembled on the assembly line
- Provide a schedule for shutting down and maintenance of the power plant
- Find identical fragments in DNA sequences

- Assign hospital wards to available locations

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft
- Provide variable values for which the logical expression is true

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft
- Provide variable values for which the logical expression is true
- Specify the order and time of advertising on radio/TV

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft
- Provide variable values for which the logical expression is true
- Specify the order and time of advertising on radio/TV
- Route the network to connect buildings in the most economical way

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft
- Provide variable values for which the logical expression is true
- Specify the order and time of advertising on radio/TV
- Route the network to connect buildings in the most economical way
- Select the stocks in which you will invest your funds

Practical examples of NP-hard problems (5/6)

- Assign hospital wards to available locations
- Make a schedule of classes
- Design the best antenna or wing of the aircraft
- Provide variable values for which the logical expression is true
- Specify the order and time of advertising on radio/TV
- Route the network to connect buildings in the most economical way
- Select the stocks in which you will invest your funds
- Assign work to employees taking into account skill constraints

Practical examples of NP-hard problems (6/6)

- "Dating site": find matching pairs

Practical examples of NP-hard problems (6/6)

- “Dating site”: find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)

Practical examples of NP-hard problems (6/6)

- “Dating site”: find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year

Practical examples of NP-hard problems (6/6)

- "Dating site": find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program

Practical examples of NP-hard problems (6/6)

- “Dating site”: find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program
- Detect the minimum set of differences in two texts

Practical examples of NP-hard problems (6/6)

- "Dating site": find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program
- Detect the minimum set of differences in two texts
- Simplify logical or algebraic expressions

Practical examples of NP-hard problems (6/6)

- "Dating site": find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program
- Detect the minimum set of differences in two texts
- Simplify logical or algebraic expressions
- Recruitment: assign candidates to specializations, taking into account their preferences

Practical examples of NP-hard problems (6/6)

- “Dating site”: find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program
- Detect the minimum set of differences in two texts
- Simplify logical or algebraic expressions
- Recruitment: assign candidates to specializations, taking into account their preferences
- Adjust weights in the neural network (“train” it)

Practical examples of NP-hard problems (6/6)

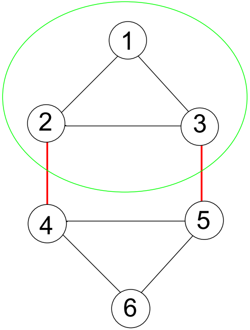
- "Dating site": find matching pairs
- Specify the flight trajectory of satellites (best photo coverage of Earth)
- Decide how much and what types of grain to grow in a given year
- Create a chess program
- Detect the minimum set of differences in two texts
- Simplify logical or algebraic expressions
- Recruitment: assign candidates to specializations, taking into account their preferences
- Adjust weights in the neural network ("train" it)
- Provide a proof of a theorem

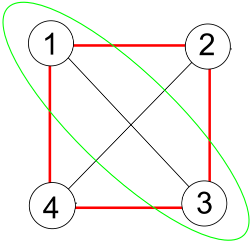
Discussion

Choose a few of the problems listed on the slides 3/6, 4/6, and 5/6, and specify

- what constitutes the set of solutions,
- how big and how limited it is,
- how to enumerate all of its elements,
- what are the evaluation criteria, and
- how to automatically calculate their values for every possible solution.

Graph bisection problem (2-GPP)





Applications:
VLSI design, networks design, data mining, geographical information systems, job scheduling

Knapsack problem

- A set of I items
- Each item $i = 1, \ldots, I$ has weight w_i and value c_i
- The backpack with a maximum weight capacity W should contain items with the maximum total value and the total weight not exceeding W

- Example interpretation – selection of investment projects
  - Items – investment projects
  - Weights – project costs
  - Values – profits from projects

Common Problems

- General function optimization
- Traveling Salesman Problem (TSP)
- Quadratic Assignment Problem (QAP)
- Graph Coloring & Partitioning (GPP)
- Minimum Spanning Tree Problem (MSTP)
- Vehicle Routing (VRP)
- Single & Multiple Knapsack
- Set Partitioning (SPP) & Set Covering Problems (SCP)
- Cutting stock problem (CSTP)
- 2-Dimensional Packing Problem (2PP)
- Processor Allocation Problem
- Staff Scheduling Problems
- Job Shop & Project Scheduling (PSP)

Compendium of NP optimization problems

https://www.csc.kth.se/tcs/compendium/

- Graph Theory
  - Covering and Partitioning
  - Subgraphs and Supergraphs
  - Vertex Ordering
  - Iso- and Other Morphisms
- Network Design
  - Spanning Trees
  - Cuts and Connectivity
  - Routing Problems
  - Flow Problems
- Sets and Partitions
  - Covering, Hitting and Splitting
  - Weighted Set Problems
- Storage and Retrieval
  - Data Storage
  - Compression and Representation
- Sequencing and Scheduling
  - Sequencing on One Processor
  - Multiprocessor Scheduling
  - Shop Scheduling
- Mathematical Programming
- Algebra and Number Theory
  - Solvability of Equations
- Games and Puzzles
- Logic
  - Propositional Logic
- Automata and Language Theory
  - Automata Theory
  - Formal Languages
- Program Optimization
  - Code Generation

Compendium of NP optimization problems

https://www.csc.kth.se/tcs/compendium/

- Graph Theory
  - Covering and Partitioning
  - Subgraphs and Supergraphs
  - Vertex Ordering
  - Iso- and Other Morphisms
- **Network Design**
  - **Spanning Trees**
  - **Cuts and Connectivity**
  - **Routing Problems**
  - **Flow Problems**
- Sets and Partitions
  - Covering, Hitting and Splitting
  - Weighted Set Problems
- Storage and Retrieval
  - Data Storage
  - Compression and Representation
- Sequencing and Scheduling
  - Sequencing on One Processor
  - Multiprocessor Scheduling
  - Shop Scheduling
- Mathematical Programming
- Algebra and Number Theory
  - Solvability of Equations
- Games and Puzzles
- Logic
  - Propositional Logic
- Automata and Language Theory
  - Automata Theory
  - Formal Languages
- Program Optimization
  - Code Generation

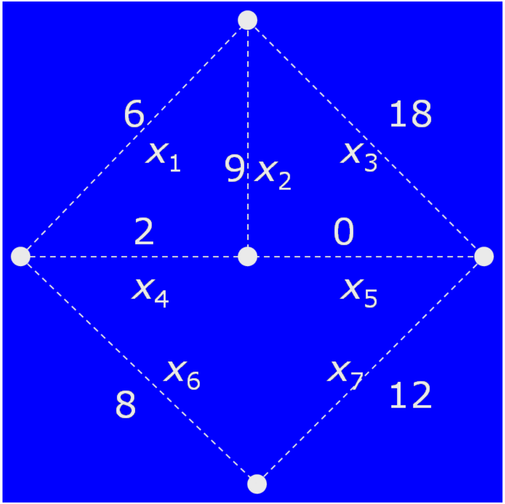
Network design NP problems

- Spanning Trees
  - MIN K-SPANNING TREE
  - MIN DEGREE SPANNING TREE
  - MIN GEOMETRIC 3-DEGREE SPANNING TREE
  - MAX LEAF SPANNING TREE
  - MAX MIN METRIC K-SPANNING TREE
  - MIN DIAMETER SPANNING SUBGRAPH
  - MIN COMMUNICATION COST SPANNING TREE
  - MIN STEINER TREE
  - MIN GEOMETRIC STEINER TREE
  - MIN GENERALIZED STEINER NET
  - MIN ROUTING TREE CONGESTION
  - MAX MIN SPANNING TREE DELETING K EDGES
  - MIN UPGRADING SPANNING TREE
- Routing Problems
  - MIN TRAVELING SALESPERSON
  - MIN METRIC TRAVELING SALESPERSON PROBLEM
  - MIN GEOMETRIC TRAVELING SALESPERSON
  - MIN METRIC TRAVELING K-SALESPERSON PROBLEM
  - MIN METRIC BOTTLENECK WANDERING SALESPERSON PROBLEM
  - MIN CHINESE POSTMAN FOR MIXED GRAPHS
  - MIN K-CHINESE POSTMAN PROBLEM
  - MIN STACKER CRANE PROBLEM
  - MIN K-STACKER CRANE PROBLEM
  - MIN GENERAL ROUTING
  - LONGEST PATH
  - SHORTEST WEIGHT-CONSTRAINED PATH
  - MIN RECTILINEAR GLOBAL ROUTING
  - MIN TRAVELING REPAIRMAN
  - MAX QUADRATIC ASSIGN
- Cuts and Connectivity
  - MAX CUT
  - MIN CROSSING NUMBER
  - MAX DIRECTED CUT
  - MAX K-CUT
  - MIN NET INHIBITION ON PLANAR GRAPHS
  - MIN K-CUT
  - MIN VERTEX K-CUT
  - MIN MULTIWAY CUT
  - MIN MULTI-CUT
  - MIN RATIO-CUT
  - MIN B-BALANCED CUT
  - MIN B-VERTEX SEPARATOR
  - MIN QUOTIENT CUT
  - MIN K-VERTEX CONNECTED SUBGRAPH
  - MIN K-EDGE CONNECTED SUBGRAPH
  - MIN BICONNECTIVITY AUGMENTATION
  - MIN STRONG CONNECTIVITY AUGMENTATION
  - MIN BOUNDED DIAMETER AUGMENTATION
- Flow Problems
  - MAX PRIORITY FLOW
  - MAX INTEGRAL K-MULTICOMMODITY FLOW ON TREES
  - MAX DISJOINT CONNECTING PATHS
  - MIN MAX DISJOINT CONNECTING PATHS
  - MIN SINGLE-SINK EDGE INSTALLATION
  - MIN UNSPLITTABLE FLOW

…And even more network design NP problems…

- Miscellaneous
  - MIN BROADCAST TIME
  - MIN K-CENTER
  - MIN K-CLUSTERING
  - MIN K-CLUSTERING SUM
  - MIN K-SUPPLIER
  - MIN K-MEDIAN
  - MIN DIAMETERS DECOMPOSITION
  - MAX K-FACILITY DISPERSION
  - MIN FACILITY LOCATION
  - MAX K-FACILITY LOCATION
  - MIN K-SWITCHING NET
  - MIN BEND NUMBER
  - MIN LENGTH TRIANGULATION
  - MIN SEPARATING SUBDIVISION

Minimal spanning tree (MST) problem

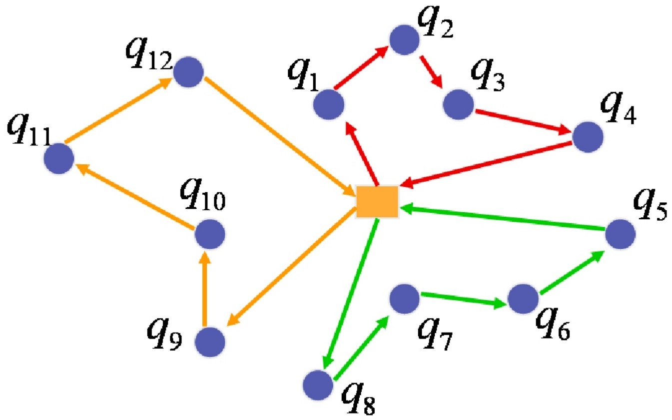


- Total cost of the links used is a minimum
- All the points are connected together

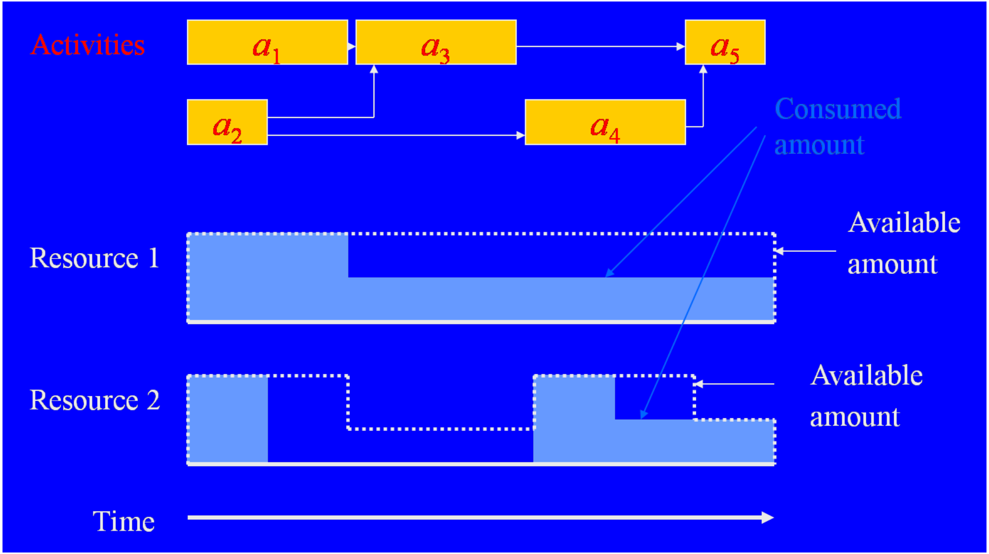
Constraint:
$x_1 + x_2 + x_6 \leq 1$
$x_1 \leq x_3$

Penalty: 50

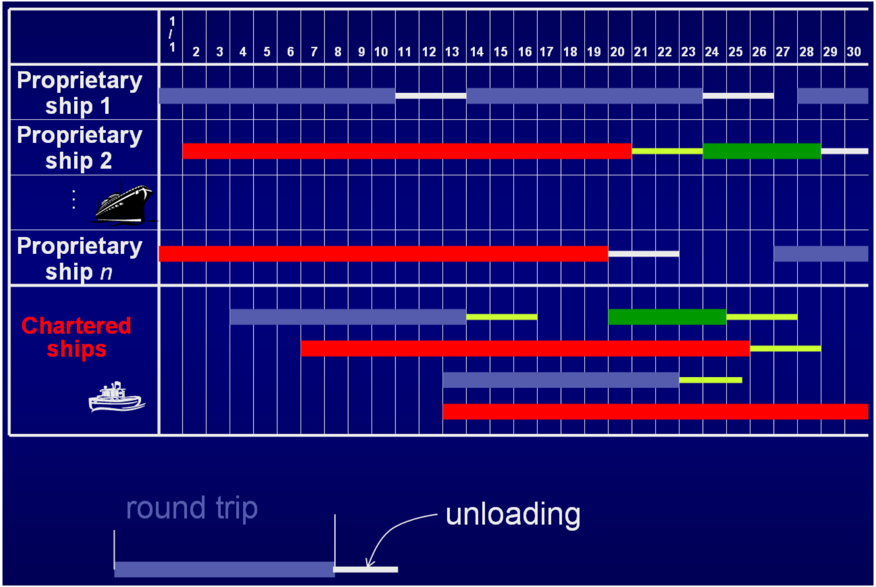
Vehicle routing problem (VRP) with constraints



capacity and **time window** constraints!

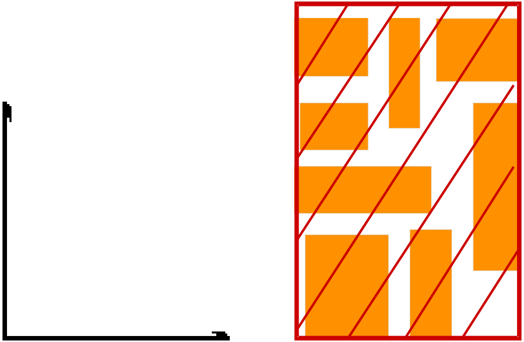

Activities
a_1
a_3
a_5
a_2
a_4
Consumed amount
Available amount
Resource 1
Resource 2
Available amount
Time

Ship schedule table

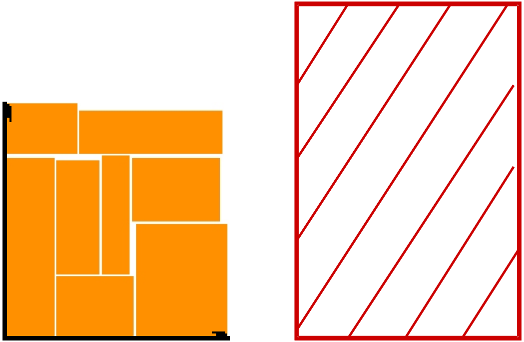


2-dimensional packing problem (2PP)

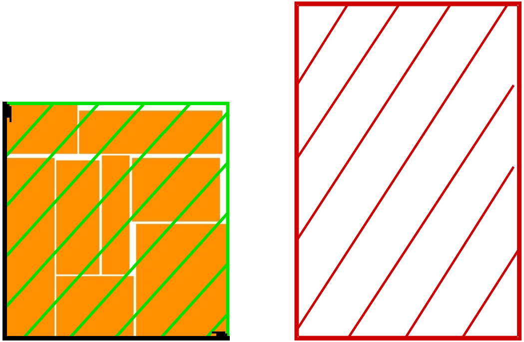
2-dimensional packing problem (2PP)



2-dimensional packing problem (2PP)

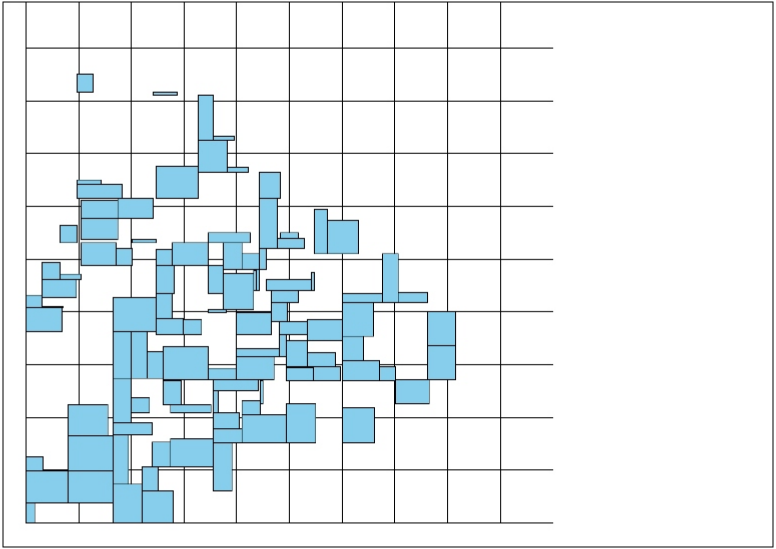


2-dimensional packing problem (2PP)

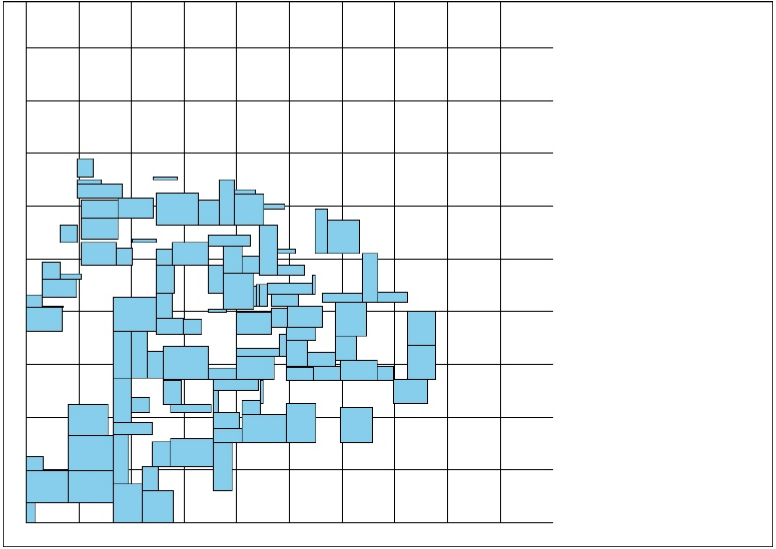


2-dimensional packing problem (2PP)

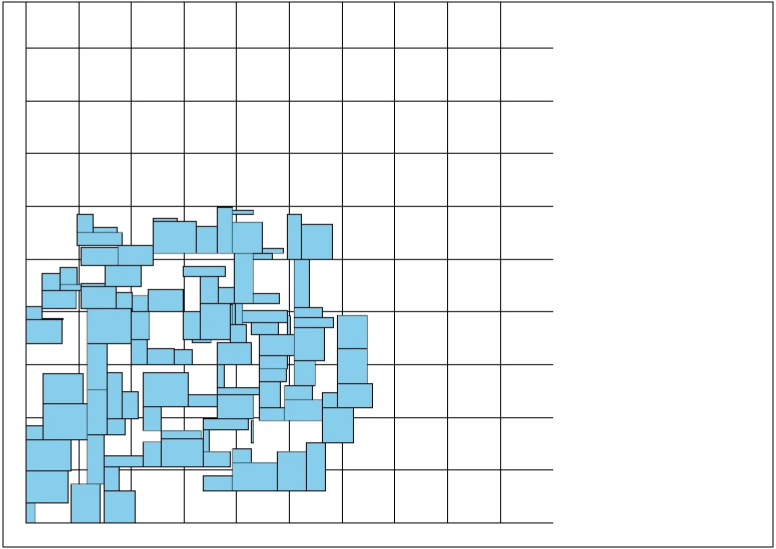
- Motivation
- Applications
- **Popular optimization problems**
- Landscapes
- Algorithms
- On-site experiments



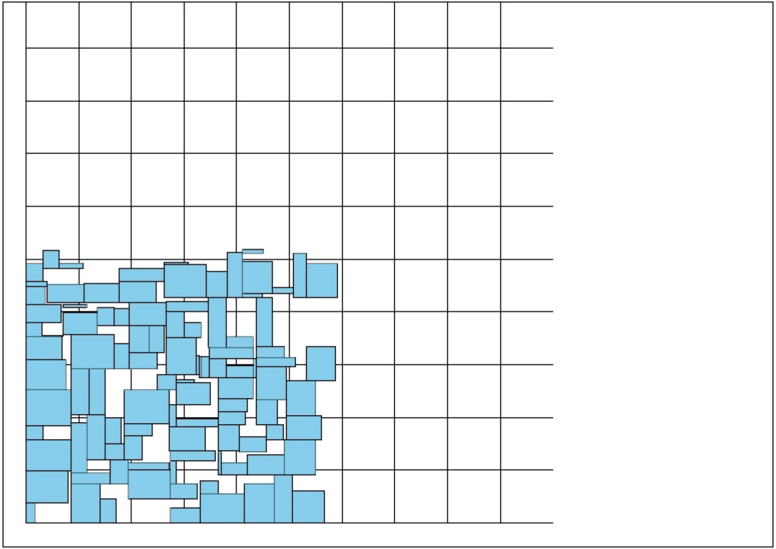
2-dimensional packing problem (2PP)



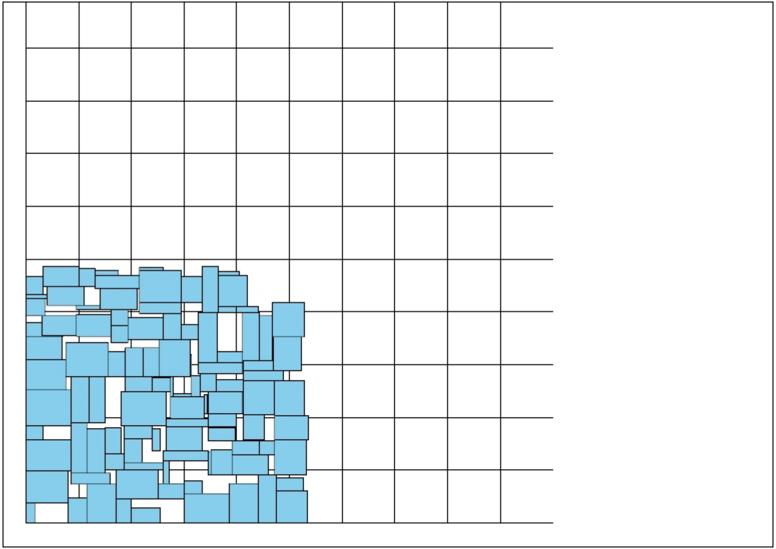
2-dimensional packing problem (2PP)



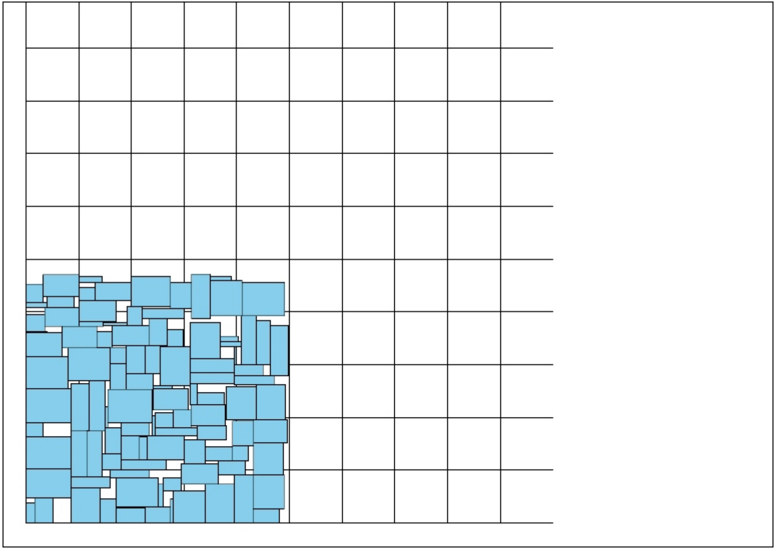
2-dimensional packing problem (2PP)



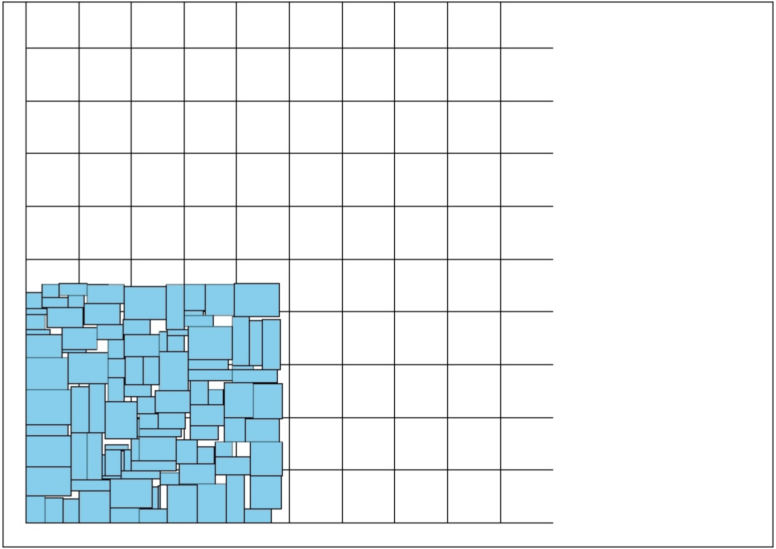
2-dimensional packing problem (2PP)



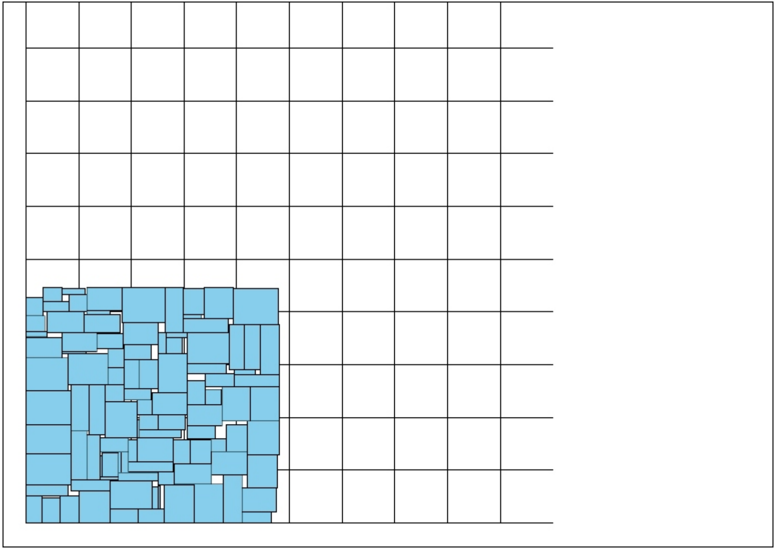
2-dimensional packing problem (2PP)



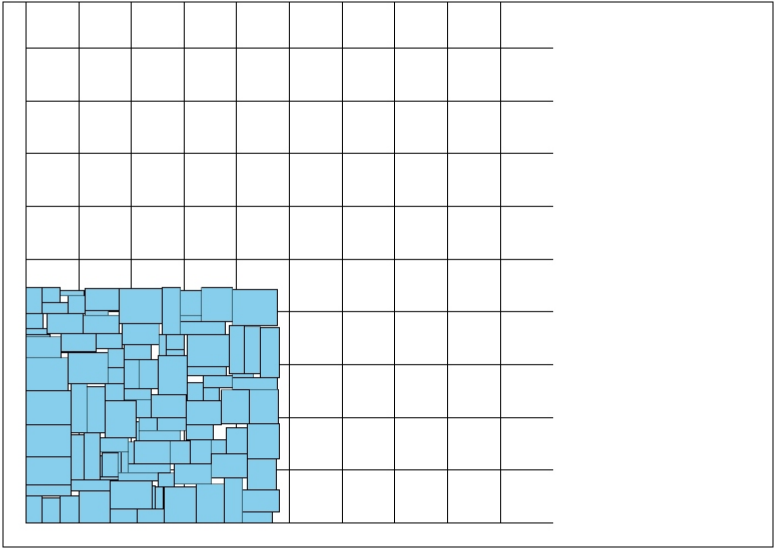
2-dimensional packing problem (2PP)



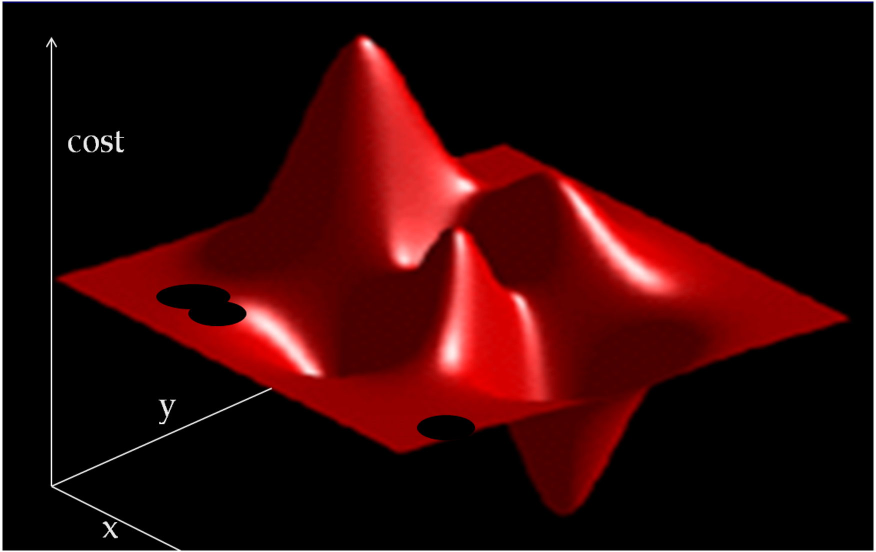
2-dimensional packing problem (2PP)



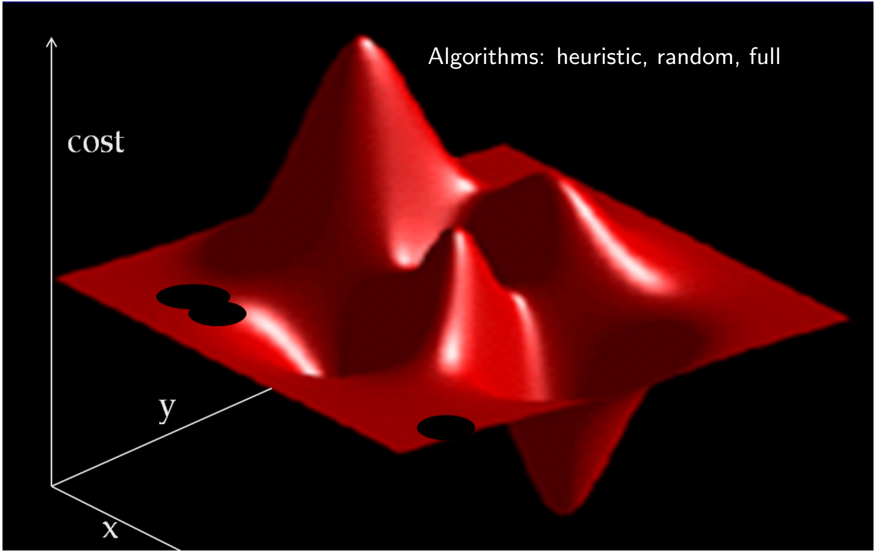
2-dimensional packing problem (2PP)



Optimization landscape

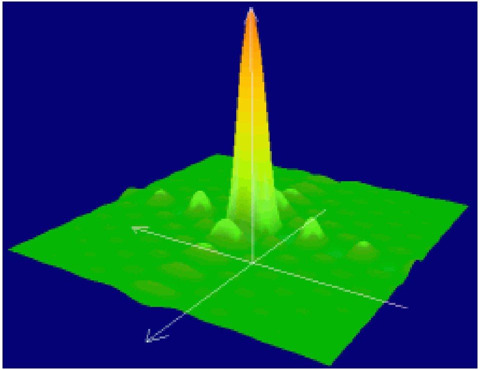


Optimization landscape



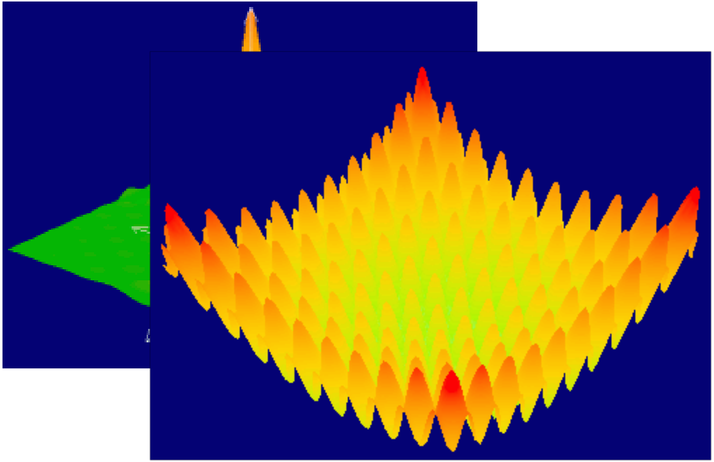
Some landscapes

http://en.alife.pl/opt/e/index.html



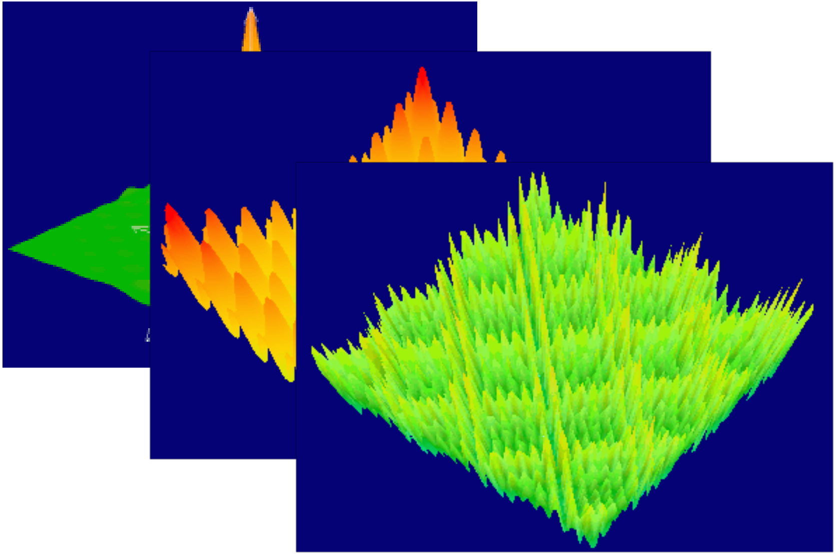
Some landscapes

http://en.alife.pl/opt/e/index.html



Some landscapes

http://en.alife.pl/opt/e/index.html



Local search

current := random solution
repeat
{
    generate a neighbor of *current*
    if the neighbor is better than *current* **then** *current* := neighbor
} **until** all neighbors are worse than *current*

Local search

current := random solution
repeat
{
    generate a neighbor of *current*
    if the neighbor is better than *current* **then** *current* := neighbor
} **until** all neighbors are worse than *current*

http://en.alife.pl/files/opt/d/OptiVisJS/OptiVisJS.html?lang=en

Evolutionary/genetic algorithm: the main loop

```
t := 0
initialize P(t)                        // P is a population (a set of solutions)
evaluate P(t)
while (not stopping-condition)
{
    t := t + 1
    select P(t) from P(t − 1)          // reproduce better solutions
    modify P(t)                        // crossover, mutate
    evaluate P(t)
}
```

Evolutionary/genetic algorithm: the main loop


```
t := 0
initialize P(t)                         // P is a population (a set of solutions)
evaluate P(t)
while (not stopping-condition)
{
      t := t + 1
      select P(t) from P(t − 1)         // reproduce better solutions
      modify P(t)                       // crossover, mutate
      evaluate P(t)
}
```

http://en.alife.pl/files/opt/d/OptiVisJS/OptiVisJS.html?lang=en

Verifying questions revisited

- Can optimization be used for **maximization**, or only for minimization?
- Are optimization algorithms used in **machine learning**?
- What is the difference between **numerical** and **combinatorial** optimization problems?
- What is a **fitness landscape**?
- What is **NP-hard**?
- What is **local search**?
- What are the main steps in an **evolutionary algorithm**?
- What is the role of **mutation** and **crossover** in evolutionary algorithms?

Hardware and software for on-site experiments

- Will you have a laptop with you?
- Linux/macOS/Windows?
- Android or iOS phone/tablet?